VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

APLIKACE S REGULÁTOREM HAWK
AN APPLICATION WITH HAWK CONTROLLER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Jaroslav Klíma**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Miroslav Jirgl, Ph.D.**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Automatizační a měřicí technika**
Ústav automatizace a měřicí techniky

Student: Jaroslav Klíma ***ID:*** 164855
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Aplikace s regulátorem HAWK

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s regulátorem HAWK a systémem COACH AX a obě tyto komponenty popište.
2. Proveďte rešerši komunikačních možností regulátoru HAWK.
3. Zaměřte se na komunikační protokol MODBUS/TCP a popište jej.
4. Realizujte propojení regulátoru HAWK a dalšího vybraného zařízení přes MODBUS/TCP a demonstrujte funkčnost.
5. Navrhněte a realizujte demonstrační úlohu.
6. Vytvořte vizualizaci.
7. Demonstrujte funkčnost.

DOPORUČENÁ LITERATURA:

HAWK: regulátor a integrátor pro aplikace automatizace budov.

Products.centraline.com [online]. Honeywell, 2015 [cit. 2017-01-30]. Dostupné z:

http://products.centraline.com/cz/ecatdata/pg_clhawk.html

Termín zadání: 6.2.2017 ***Termín odevzdání:*** 29.5.2017

Vedoucí práce: Ing. Miroslav Jirgl, Ph.D.
Konzultant:

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

ABSTRAKT

Cílem této bakalářské práce je realizace propojení regulátoru HAWK od společnosti Centraline/Honeywell a modelů jeřábů přes komunikační protokol MODBUS/TCP, včetně demonstrace funkčnosti a vytvoření vizualizace. Pro řízení modelů byly použity programovatelné automaty SIEMENS S7-1200. První část práce byla věnována popisu HAWK regulátoru a jeho komunikačních možností v základní konfiguraci. Druhá polovina teoretické části se zabývá nejpoužívanějšími funkcemi a terminologií vývojového prostředí COACH^{AX}, pomocí jehož je realizována MODBUS komunikace a vytvořena demonstrační aplikace. Praktická část nabízí jednak jednoduchou úlohu pro demonstraci komunikace programovatelného automatu a regulátoru HAWK přes MODBUS/TCP a dále aplikaci, která vytváří nadstavbovou systémovou vrstvu, oproti programovatelným automatům, a nabízí pokročilé možnosti ovládání modelu a vytvoření základních provozních statistik.

KLÍČOVÁ SLOVA

HAWK, COACH^{AX}, Centraline, Honeywell, vizualizace, sběr dat, PLC, MODBUS, TIA PORTAL

ABSTRACT

The target of this bachelor's thesis is to execute a connection between the HAWK controller by the Centraline/Honeywell company and crane models through the MODBUS/TCP communication protocol, including a demonstration of its functionality and creating a visualisation. Programmable machines SIEMENS S7-1200 were used to control the models. The first part of the thesis is dedicated to describing the HAWK controller and its communication options in its basic configuration. The second part is dedicated to the most utilized functions and terminology of the COACH^{AX} development environment thanks to which the MODBUS communication is executed and a demonstration application is created. The practical part first offers a basic exercise to demonstrate the communication of a programmable machine and the HAWK controller through MODBUS/TCP and then an application which creates a shell system surface, unlike the programmable machines, and offers advanced options of controlling the model and formation of basic functional statistics.

KEYWORDS

HAWK, COACH^{AX}, Centraline, Honeywell, visualization, data collection, PLC, MODBUS, TIA PORTAL

Bibliografická citace:

KLÍMA, J. *Aplikace s regulátorem HAWK*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2017. 65s. Vedoucí práce: Ing. Miroslav Jirgl, Ph.D.

PROHLÁŠENÍ

„Prohlašuji, že svou závěrečnou práci na téma Aplikace s regulátorem HAWK jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne **29. května 2017** ……

podpis autora

PODĚKOVÁNÍ

Děkuji vedoucímu bakalářské práce Ing. Miroslavu Jirglovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne **29. května 2017** ……

podpis autora

OBSAH

1 Úvod ... 13
2 Technologie a systémy pro řízení a analýzu dat ... 14
2.1 HAWK regulátor ... 14
2.2 Programovací rozhraní COACH^{AX} ... 16
2.3 Popis služeb a aplikací v prostředí COACH^{AX} ... 18
2.3.1 Datové typy ... 18
2.3.2 Funkční bloky ... 18
2.3.3 Knihovny ... 18
2.3.4 Alarmy ... 19
2.3.5 Sledování změn proměnných ... 19
2.3.6 Softwarové nastavení I/O v HAWK ... 19
2.4 Navigace ... 20
2.4.1 Vývojářské režimy / pohledy ... 21
2.4.2 Založení nového projektu ... 21
2.5 Komunikační protokoly ... 21
2.5.1 LonWorks ... 22
2.5.2 BACnet ... 22
2.5.3 Sběrnice KNX/EIB ... 24
2.5.4 Sběrnice M-BUS ... 24
2.5.5 Protokol OPC ... 25
3 Komunikační protokol MODBUS ... 25
3.1.1 Popis komunikačního protokolu MODBUS ... 26
3.1.2 MODBUS na sériové lince (RS-232, RS-485) ... 27
3.1.3 MODBUS RTU ... 28
3.1.4 MODBUS ASCII ... 29
3.1.5 MODBUS TCP/IP ... 29
4 Propojení HAWK s reálným modelem přes MODBUS TCP ... 30
4.1 Popis PLC modelu ... 30
4.1.1 Ovládání pohonů ... 31
4.1.2 Prostředí TIA PORTAL ... 31
4.2 Popis testovací úlohy ... 32
4.3 MODBUS konfigurace PLC modelu ... 32
4.4 MODBUS konfigurace na HAWK regulátoru skrze COACH^{AX} ... 33
4.4.1 Použitý HAWK integrátor a prostředí COACH^{AX} ... 34
4.5 Demonstrace funkčnosti na změně stavu výstupu ... 35
5 Návrh a realizace praktické úlohy pro HAWK regulátor ... 37
5.1 Konfigurace sítě ... 39

5.2 Stavy a hodnoty proměnných z PLC a jejich přenesení do prostředí COACH^{AX} ... 40
5.2.1 Přenesení dat na pracovní plochu ... 41
5.3 Ovládací prvky v prostředí COACH^{AX} ... 41
5.4 Přístup k uživateli vytvořeným souborům ... 42
5.5 Logika spojená s ovládáním a vizualizací ... 42
5.5.1 Standardizace rychlosti hlavního motoru ... 43
5.5.2 Přístup k webovému prostředí ... 44
5.6 Vizualizace pomocí webového prostředí ... 45
5.7 Popis vytvořené aplikace ... 46
5.7.1 Blok 1 - Okamžité hodnoty a stavy proměnných ... 46
5.7.2 Blok 2 - Vizualizace ... 48
5.7.3 Blok 3 – Grafické vyjádření modelů ... 50
5.7.4 Blok 4 – Doplňující informace ... 51
6 Závěr ... 53
LITERATURA ... 55
SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK ... 58
SEZNAM PŘÍLOH ... 59

SEZNAM OBRÁZKŮ

Obrázek 1 - HAWK regulátor/integrátor [2] 16
Obrázek 2 - Prostředí COACH^{AX}, správa pointů (kapitola 2.3.6 a 4.4) 17
Obrázek 3 - Náhled do navigace prostředí COACH^{AX} 20
Obrázek 4 - LON karta PCI a LON USB adaptér – U10 [22], [23] 22
Obrázek 5 - Model objektu snímače teploty v BACnetu [11] 23
Obrázek 6 - Přehled základních objektů v BACnetu [11] 23
Obrázek 7 - Přenos dat bez OPC (vlevo) a přenos dat pomocí OPC (vpravo) [21] 25
Obrázek 8 - Základní tvar MODBUS zprávy 26
Obrázek 9 - MODBUS provedení bezchybného požadavku 26
Obrázek 10 - MODBUS provedení chybného požadavku 27
Obrázek 11 - MODBUS zpráva na sériové lince 28
Obrázek 12 - MODBUS zpráva v režimu RTU 28
Obrázek 13 - Formát bytu u režimu MODBUS RTU 28
Obrázek 14 - Zpráva v režimu ASCII 29
Obrázek 15 - Formát bytu u režimu MODBUS ASCII 29
Obrázek 16 - Základní tvar MODBUS zprávy na TCP/IP 29
Obrázek 17 - PLC model využitý při demonstraci MODBSU TCPCP/IP komunikace a realizaci praktické úlohy [24] 31
Obrázek 18 - Blokové schéma zapojení při demonstraci komunikaci přes MODBUS TCP/IP 32
Obrázek 19 - Nastavení PLC modelu jako MODBUS server v prostředí TIA PORTAL V13 33
Obrázek 20 - Konfigurace server MODBUS zařízení v prostředí COACH^{AX} 34
Obrázek 21 - Hodnoty a stavy proměnných před změnou v ovládacím prostředí COACH^{AX} 35
Obrázek 22 - Vizualizace ovládacího panelu pro změnu výstupní a pomocné hodnoty v PLC 36
Obrázek 23 - Hodnoty a stavy proměnných po změně v ovládacím prostředí COACH^{AX} 36
Obrázek 24 - Náhled do MODBUS sítě při realizaci praktické úlohy 39
Obrázek 25 – Blokové schéma zapojení pracoviště pro realizaci praktické části úlohy 39
Obrázek 26 - Nastavování parametrů bodu (pointu) čidlo_1 typu boolean 40
Obrázek 27 - Virtuální tlačítko (pomoc3) pro start hlavního motoru 43
Obrázek 28 - Kompletní vytvořená vizualizace skrze webové rozhraní 45
Obrázek 29 - Blok 1) Náhled na část vizualizace obsahující aktuální stavy proměnných 46

Obrázek 30 - Vývojový diagram pro identifikaci stavu modelu 47
Obrázek 31 – Blok 2) Vizualizovaný model jeřábu včetně ovládacích prvků 48
Obrázek 32 - Vývojový diagram pro selektování stavu modelu jeřábu při vizualizaci 49
Obrázek 33 - Blok 3) Časové závislosti nejsledovanějších veličin z reálných modelů 50
Obrázek 34 - Blok 4) Doplňující informace k vizualizaci, komunikaci s modely a regulátorem HAWK 51

SEZNAM TABULEK

Tabulka 1 – Datový model MODBUS [12] 30
Tabulka 2 - Seznam činností vytvořeného nadřazeného systému 38

1 ÚVOD

V dnešní moderní době informačních technologií je snaha o postupnou automatizaci mnoho aktivit. Máme snahu o nahrazení činnosti, ale zároveň ponechat přehled o veškerém stavu zařízení. Pro tyto účely slouží nejrůznější řídicí a informační systémy. Jedním typem takových systémů jsou tzv. integrátory.

Ty jsou schopny sbírat data z nejrůznějších periférií objektu a převést je na člověkem čitelný výstup. Obrovská výhoda a hlavní důvod použití tohoto prvku je, že každá jednotka může být schopna komunikovat rozdílným protokolem a než řešit problém převodníkem, zvolíme raději možnost integrátoru. Příkladem takového integrátoru je zařízení HAWK od společnosti Centraline/Honeywell. Zařízení HAWK je schopno v základní konfiguraci komunikovat pomocí několika nejpoužívanějších protokolů a v případě kolize kompatibility je možné přidat protokoly pomocí externích karet, a tak data sloučit do jedné platformy.

V dalších kapitolách práce je věnovaný prostor popisu regulátoru/integrátoru HAWK, který je pomocí praktické úlohy prezentován. Výsledný výstup práce je vizualizován pomocí webového rozhraní, včetně modelů a stavů periférií s minimální odezvou. Vše za pomocí web serveru, který obsahuje integrátor v základní konfiguraci. A jeho hlavním úkolem bude sbírat data z dostupných reálných modelů a jejich výsledný model vizualizovat.

Jedná o nadřazený systém, který může ovlivňovat chod modelů, např. pomocí nulování čítačů nebo nastavení limitu aktivovaných senzorů. Tedy takový rozsah práce, na který obsluha u modelu jeřábu nemá příslušné oprávnění.

2 TECHNOLOGIE A SYSTÉMY PRO ŘÍZENÍ A ANALÝZU DAT

Řídicí systém získává data ze zdrojů (např. senzorů) a na základě nahraného programu v paměti centrální jednotky lze měnit jejich hodnoty a tím řídit celý systém. Řídicí systém je zařízení obsahující regulátor a vstupní/výstupní moduly. Právě díky modulům dokáže systém komunikovat. Signály skrze moduly mohou být analogové nebo digitální a řídicí systém může skrze ně komunikovat po mnoha dostupných protokolech.

Analogové – vstupní hodnota je analogová, která může být elektrická, tak neelektrická. Hodnota je v rozsahu 0-10 V [5]. Vstupním parametrem je neznámé číslo, které je potřeba standardizovat. To znamená, že na výstupu analogového potenciometru nebudou obsluhou nečitelná čísla, ale optimalizované vhodné hodnoty.

Digitální – hodnoty digitální jsou prezentovány 1 nebo 0. V řídicím pojmu lze snadněji pochopit jako zapnuto nebo vypnuto. Kontakt může být napěťový nebo bezpotenciální. Digitální výstupy vysílají hodnoty bitově, tím pádem ovládají spínací objekty (on/off).

Dnešní řídicí systémy nabízejí uživatelské rozhraní „web server“, který lze integrovat v podobě web stránky ve webovém prohlížeči a celý proces řízení sledovat pomocí vizualizace z operátorsky příjemnějšího místa[1]. Prakticky může osoba celý proces, tzn. Stavy proměnných a tím pádem stav senzorů nebo požadované hodnoty sledovat a přizpůsobovat.

V mé práci se budu věnovat konkrétně řídicímu systému od firmy Centraline/Honeywell a tím je systém HAWK, na který vytvořím aplikaci vizualizovanou přes webové rozhraní.

2.1 HAWK regulátor

Často označovaný jako integrátor či regulátor je integrovaná řídicí jednotka pro řízení technologií, sledování dat, záznam dat, zálohování dat, alarmování a funkce pro správu sítě s internetovým připojením a webovým serverem (HAWK obsahuje aplikaci web server) skrze uživatelsky příjemné prostředí. Týkající se např. vytápění, větrání a klimatizace – tzv. HVAC, dále také ovládání osvětlení. Sledování spotřeby energie či další aplikace z oblasti inteligentních budov nebo průmyslových zařízení) [2]. Z toho vychází jedna z jeho největších výhod, že může přijímat data z různých komunikačních sítí. Ty se mohou lišit v protokolech nebo architekturách a v příslušném inženýrském nástroji vytvoříme síť zahrnující požadovaný komunikační protokol (sítí může být více) [15].

Výhodou je kromě řízení a ovládání přes web možnost přístupu k aktuálním informacím pomocí webového prostředí (skrze webový prohlížeč) [2]. HAWK umožňuje chytré plánování a široké projekční možnosti pro projekty jakýkoliv velikostí.

Nabízí možnost spravovat externí zařízení skrze internet a přenášet uživateli informace s minimální odezvou. Je navržený pro jednoduchou správu a v základní konfiguraci komunikuje na množství rozšířených protokolů, kterými jsou:

- LON (plug-in karta CLAXHAWKIFLON)
- BACnet IP, MSTP
- OPC
- M-Bus RS232 a M-Bus master
- ModBus Asynchronně, Slave, TCP
- EIB IP

Rozšiřující ovladače dále jsou [1]:

- CLAXDREVO Systém Honeywell Evo Home
- CLAXDRBPORT C-Bus přes ovladač Bport
- CLAXDRSMS SMS služby pro HAWK přes HSM/GPRS modem

Pro kompletní seznam je nutné vyžádat seznam prostřednictvím lokálního obchodního zastoupení Centraline[1].

Integrátor je možné pořídit ve dvou vyráběných sériích a to 300E a 600E. Verze se liší v rychlosti procesoru, 300E verze nabízí frekvenci 400 MHz, zatímco verze 600E frekvenci 524 MHz[2]. Obě série navíc nabízí služby pro obnovu dat a možnost pracovat v režimu absence baterie, což je vhodné např. do prostředí s vyšší teplotou vzduchu. Díky svým kompaktním rozměrům lze bezproblémově připevnit např. na DIN lištu nebo stěnu.

Pro regulátor existují dvě sady I/O modulů, HAWK disponuje přímo portem pro připojení těchto rozšíření. Prvním je IO16 (CLAXHAWKIO16), které nabízí 8 univerzálních vstupů, 4 digitální výstupy, 4 analogové výstupy a možnost připojení až 4 modulů na regulátor. Zatímco verze IO34 (CLAXHAWKIO34) disponuje 16 univerzálními vstupy, 8 analogových výstupů, 10 digitálních výstupů, integrované napájení 24 V AC/DC, avšak možnost připojení pouze jednoho modulu na regulátor[7]. Pod pojmem univerzálních vstupů si lze představit signál o napětí 0 – 10V nebo proud 4 – 20 mA. Lze také vstupem definovat nelineární charakteristiku nebo připojit teplotní snímač (charakteristika 10 kΩ).

Analogové hodnoty portů se pohybují 0 – 10 VDC. Digitální maximálně 30 V AC/DC a proud 0,5 A. Digitální porty dosahují pouze stavů zapnuto / vypnuto.

Komunikační rozhraní obsahuje integrátor v základní konfiguraci a jsou to 2 ethernetové porty (rychlost 10/100 MB, postačuje základní síťová karta v PC, konektor RJ-45), 1x RS232 port (9-PIN D) a 1x RS485 (3 svorky) [4].
V případě omezení lze porty rozšířit o přídavné karty, HAWK obsahuje 2 sloty pro přídavné karty. Seznam základních přídavných karet:

- LonWorks port (CLAXHAWKIFLON)
- 2x RS485 (CLAXHAWKIF485)
- RS232 (CLAXHAWKIF232)

Ve výsledku lze tedy zapojit 66 datových bodů[2].

Obrázek 1 - HAWK regulátor/integrátor [2]

2.2 Programovací rozhraní COACH^{AX}

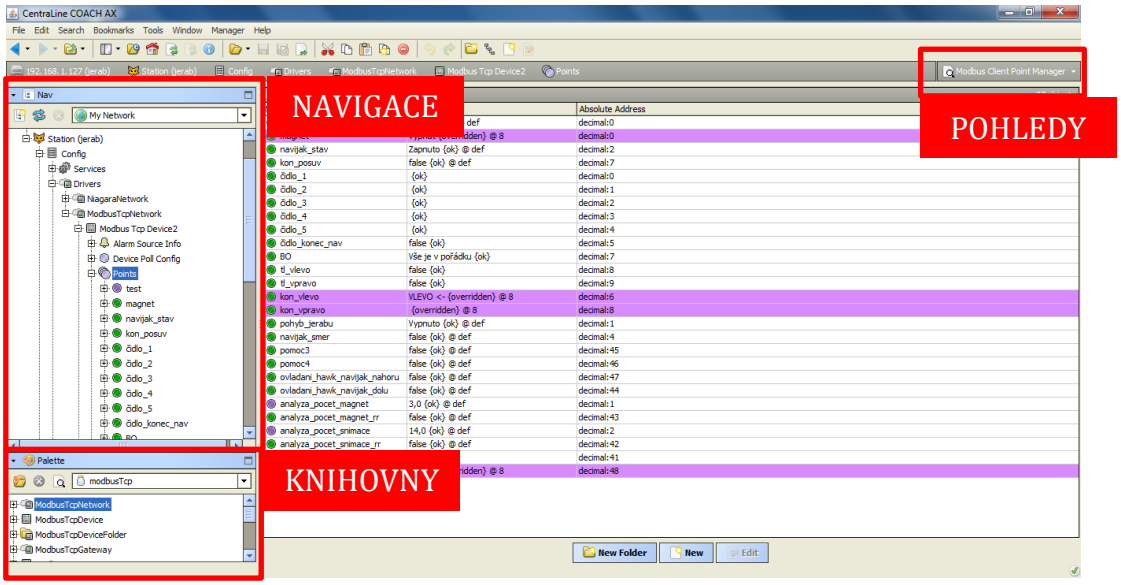
Inženýrský nástroj COACH^{AX} je programový nástroj, vývojové prostředí, od firmy Centraline/Honeywell spojené zejména s moderním řízením budovy, ale i dalšími aplikacemi, sloužícími pro programování řídicích aplikací a konfiguraci pro HAWK regulátor a sledovací systém ARENA^{AX}. Představuje jakýsi koncept softwarového prostředí sjednocující data, čtení stavu proměnných, vytváření řídicí logiky, nastavování alarmů a tvorbu grafické vizualizace systému bez ohledu na výrobce nebo komunikační protokol. Systém není klasickým vývojovým prostředím, ale základem je grafické programování spočívající v přesouvání prvků na pracovní plochu a tam definovat vzájemný stav grafickým propojením prvků nebo nastavením stavu proměnné. Při změně parametrů prvků se změny projeví téměř okamžitě.

Sběr informací z různých zařízení od různých výrobců do jedné aplikace je časové náročný a drahý. Prostředí COACH^{AX} umožňuje právě tuto implementaci, seznam nabízených funkcí je:

- Připojení zařízení různých komunikačních protokolů
- Otevřené API
- Nezávislost na výrobci
- Přístup skrze webové rozhraní
- Rozsáhlá sada knihoven (možnost doprogramovat modul)

Software pro funkčnost vyžaduje přítomnost stanice a platformy. Výhodou prostředí je schopnost práce na různých platformách. Platformou se rozumí fyzické zařízení jako PC, na kterém je software nainstalovaný nebo samotný HAWK. COACH^{AX} dokážeme spustit na různých HW zařízení a operačních systémech. Platforma se stará o služby zapojených stanic. Rozhoduje se, které funkce u stanic budou aktivní. Přihlášení do platformy je závislé na jejím typu, do HAWK integrátoru se přihlašujeme pomocí přihlašovacích údajů (dodaných spolu s regulátorem) a v případě PC jsou přihlašovacími údaji administrátorské údaje (přihlášení do operačního systému) [13].

Ve stanici je obsažena aplikace, vytvářející se v grafickém prostředí. V COACH^{AX} je možné navrhnout velký počet stanic (záleží na komunikačním protokolu), avšak aktivní může být pouze jedna. Do regulátoru lze nahrát pouze jednu stanici. Přihlašovací údaje do stanic lze přizpůsobovat obrazu svému díky službě *Category service*.



Obrázek 2 - Prostředí COACH^{AX}, správa pointů (kapitola 2.3.6 a 4.4)

2.3 Popis služeb a aplikací v prostředí COACH^{AX}

Prostředí obsahuje řídicí jádro provádějící zvolenou aplikaci a má široké množství funkcí od jednoduchého řízení po aplikace se složitými řídicími algoritmy a náročnou logiku. Systém COACH^{AX} je napsaný v jazyce JAVA, tím pádem stačí na HW platformě mít nainstalovanou potřebnou verzi JVM a je zde podpora programování vlastních rozšíření. Výchozí funkce umožňují aplikacím předávat chybové hlášení, hlásit podmíněné události koncovému uživateli, těmto službám je věnován prostor v další části práce [18], [19].

2.3.1 Datové typy

Čtyři základní datové typy podporované prostředím COACH^{AX}

- Boolean – bitové hodnoty (zelená barva)
- Numeric – celočíselný tvar (fialová barva)
- Enumerated – víceparametrový typ (oranžová barva)
- String – řetězec (bílá barva)

Datové typy lze jednoduše přetypovat pomocí příslušných konvertorů. Datové typy můžeme nastavit také jako konstanty [6].

2.3.2 Funkční bloky

Nejpoužívanější prvky v systému COACH^{AX} jsou:

- Matematické funkce – sčítání, odčítání, průměr apod.
- Logické funkce – AND, OR, NOR apod.
- Časovače (Timer) – Prvky používané např. pro zpoždění
- Generátory – sinus, náhodné číslo, impulz
- HVAC – PID regulátor, hystereze

Všechny prvky jsou na výběr z palety nástrojů. Jelikož je prostředí řešené pomocí grafického programování, stačí vybraný prvek přesunout na pracovní plochu[18], uvedené prvky jsou součástí výchozí knihovny *kitControl*.

2.3.3 Knihovny

V prostředí jsou k dispozici různé předpřipravené grafické prvky jako například vzduchová šachta, která by se mohla uplatnit při tvoření systému, kde je klíčovým objektem klimatizace. Jsou zde různé prvky od identifikačních žárovek po potrubí. V prostředí jsou pod tzv. *Palette*, knihovny je možné updatovat, instalovat z neoficiálních zdrojů a mazat v nastavení platformy, konkrétně

větvi *Software Manager*. Abychom mohli využívat grafické prvky, je nutné do palety přidat knihovnu *kitControl*. Prostředí musí být nastaveno na *Graphic* pohled.

2.3.4 Alarmy

Alarmy lze připojit k prvku za účelem sledování stavu. Identifikaci je možné aktivovat při nastavení určitého intervalu výstupní hodnoty alarmu. Pro zobrazení alarmu je nutné využít konzolového příjemce. Dokonce je možné nastavit odesílání alarmů pomocí emailu (např. jako bezpečnostní prvek) [20].

2.3.5 Sledování změn proměnných

Prostředí podporuje možnost sledovat změny stavů proměnných a exportovat (např. do grafu) do různých formátů jako PDF, HTML a CSV. Dále je možné snímat hodnotu v určitém intervalu hodnot nebo při každé změně.

Blok, který umožňuje sledovat proměnné, se nazývá *extension* k nalezení v knihovně *history*. Prvek se aktivuje přidáním ke sledované proměnné (nutné povolit).

2.3.6 Softwarové nastavení I/O v HAWK

K přímému použití vstupů a výstupů v regulátoru je slouží *NDIO driver*. Ovladač je nutné nainstalovat do modulu HAWK a je součástí *NDIO modulu*. K seznamu nainstalovaných modulů se dostaneme pomocí *Platform Software Manager*, kde je k dispozici seznam všech použitelných modulů. Nalezneme zde položku s názvem *NDIO* a nainstalujeme. Pro úspěšnou implementaci do regulátoru je nutné HAWK restartovat. *NDIO* ovladač přidáme za pomocí tlačítka *ADD NDIO Network* ve složce *Driver*. Ve vytvořené *NDIO* síti stiskneme tlačítko *Discover* a přidáme *NDIO Board*. Každý *NDIO Board* prezentuje jeden I/O modul (je možné připojit více).

Rozdíly modulů jsou v portu, čím menší číslo portu, tím blíže je modul k centrální jednotce. Správný postup nám identifikuje LED na přední straně modulu. Pod NDIO Bordem se nachází přídavný blok s názvem *Points*, kde po stisknutí *Discover* se zobrazí všechny použitelné vstupy a výstupy modulu.

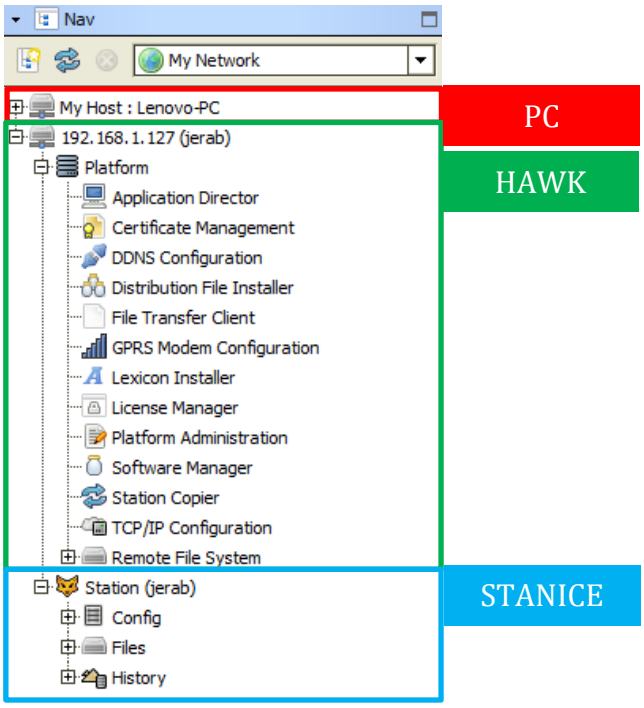
Po vytvoření vstupů a výstupů se objeví v navigačním menu pod *Points* vytvoří seznam obsahující použitelné vstupy a výstupy. Kliknutím pravým tlačítkem na danou položku je nutné zvolit *link to mark*. V grafickém prostředí přes paletu zvolíme blok *Numeric Writable* v případě analogové hodnoty *nebo BooleanWritable* pro digitální hodnotu. Dále vybereme v nabídce (pravé tlačítko) *link to* nebo *link from* v závislosti jestli se jedná o vstup nebo výstup. Po úspěšné operaci vyskočí dvě okna vedle sebe, kde je možné vybrat vzájemné propojení dle libovolného výběru. Po propojení je možné posílat hodnoty na výstupy nebo číst příslušné vstupy [13].

2.4 Navigace

Hlavní část celého vývojového prostředí je bezpochyby navigace stromové struktury v levé části obrazovky (při výchozím nastavení). Jde o jakýsi přehled připojených stanic a platforem. Platformy viditelné na obrázku 3 jsou osobní počítač (červená barva) s nainstalovaným prostředím a HAWK regulátor (zelená barva). Navigace představuje celkovou správnou nad činností stanic a platforem, například spouštění stanic, správa licencí, instalace modulů s externích zdrojů, správa souborů apod.

Na každé platformě může být aktivní pouze jedna stanice, o zvolenou aktivní stanici se stará nástroj *Application Director*, je-li stanice aktivní lze je na ní možné provádět úpravy. Při vývoji více projektů např. na lokálním počítači je možná přítomnost více stanic, opět platí pravidlo, že aktivní může být pouze jedna. HAWK může obsahovat pouze jednu stanici, kterou je ale možné editovat na lokálním počítači.

V nastavení stanice je definována komunikační síť, logika a přístupová práva. Klíčová větev s názvem *Config* obsahuje konfiguraci sítě + typy datových bodů vstupů a výstupů, dále definice uživatelů a jejich práva, dále chybové a výstražné hlášení (viz kapitola 2.3.4).



Obrázek 3 - Náhled do navigace prostředí COACH^{AX}

2.4.1 Vývojářské režimy / pohledy

Při vývoji aplikace využíváme více pohledů na projekt. Každý je specifický svojí funkcí. Režimy při vývoji v prostředí COACH^{AX} přepínáme v pravém horním rohu. V případě vizualizace, musíme vytvořit *Folder* (složku) a v ní vytvořit nový pohled režimu *Graphic*. Tímto dosáhneme prostředí grafické vizualizace, při kliknutí na *Views -> New view* pouze zvolíme název režimu a vše ve výchozím nastavení potvrdíme.

Dalšími režimy poskytující prostředí jsou například *Wire*, kde drátujeme jednotlivé body (proměnné) a možnosti nabízené aktuální knihovna jako jsou různé matematické operace, logické operace a vzájemné závislosti. Dalším využívaným režimem je *Property Sheet*, který slouží ke sledování a nastavování stavů a hodnot parametrů. Při založení nového grafického pohledu budeme mít na výběr dva grafické režimy, jeden slouží pro vývoj (před názvem *Edit*) a druhý slouží jako výsledný výstup pro testování.

2.4.2 Založení nového projektu

Při založení nového projektu je nutné nastavit platformy a stanice. Jako první krok je nutné provést vytvoření platformy, což lze brát jako páteřní jednotku sítě. Vytváří se v záhlavním menu *FILE -> open platform*. Po kliknutí na položku se objeví tabulka, kam zadáme IP adresu platformy.

Pokud aplikaci pouze simulujeme, bude IP adresa platformy lokální adresa počítače (tj. 127.0.0.1). V posledním okně jsou vyžádány přihlašovací údaje OS.
Dalším krokem pro vytvoření funkčního simulovaného modelu je vytvořit stanici (obsahuje kód, který bude prováděn). Stanici vytvoříme v menu *TOOLS -> New Station*. Prostředí si vyžádá po uživateli vyplnit název stanice a vytvoření hesla k přihlášení. Heslo musí obsahovat alespoň jedno velké písmeno, numerickou cifru a minimální počet znaků je 10.

Po aktivaci vybereme platformu a z nabídky zlovíme *Application director*. Na obrazovce uvidíme seznam všech stanic, avšak jak již bylo zmíněno v předchozích kapitolách zabývajících se prostředím COACH^{AX}, stanic může být více, ale aktivní pouze jedna. Pokud chceme přepnout stanici, musíme aktivní stanici zastavit tlačítkem STOP.

2.5 Komunikační protokoly

Komunikační protokoly zajišťují vzájemnou komunikaci mezi prvky v řízeném okruhu, přizpůsobuje propojení prvků s různými druhy sběrnic a jakou rychlostí a technologií budou vzájemné prvky komunikovat (tedy vysílat a přijímat data), Protokol dále určuje množství prvků v síti [15], [3].

2.5.1 LonWorks

Technologii vyvinula firma Echelon v letech 1989 až 1992 ve spolupráci s předními firmami jako Toshiba a Motorola, přičemž její uvedení na trh proběhlo v roce 1992. Vychází z obecné definice sítě (LON – Local operating networks). Tyto sítě jsou složeny z inteligentních zařízení a uzlů. Jsou propojeny a komunikují spolu v síti jedním komunikačním protokolem. Uzly jsou přizpůsobeny na vysílání zpráv při změně stavů nebo při reakci na přijatou zprávu.

Technologie odesílá data v závislosti na vzdálenosti. Umožnuje fyzicky připojení až 64 zařízení a mezi nimi vytvořit 32 385 sítových uzlů. U kroucených párů vodičů je maximální délka sítě činní 2 700 m, avšak při této vzdálenosti se maximální rychlost sníží přibližně na 10 kb/s. Pro zachování maximální dostupné rychlosti nabízející tento protokol, musí být délka sítě maximálně 1 500 m [3].

Technologie disponuje univerzální komunikací po libovolné komunikaci včetně RS485. Tímto si zaručuje využití např. pro dálkové odečty energií, řízení spotřebičů nebo automatizaci budovy (tzv. Inteligentní budova). Pro připojení libovolného zařízení do sběrnice LON vyžaduje převodník [6].

PC požadující komunikaci pomocí technologie LON je nutné osadit např. zásuvnou kartou do PCI slotu:

Obrázek 4 - LON karta PCI a LON USB adaptér – U10 [22], [23]

2.5.2 BACnet

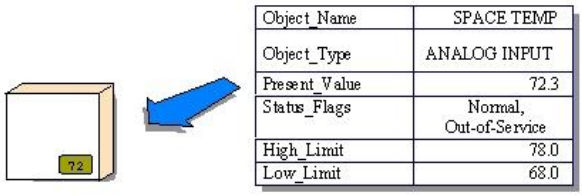
Jedná se o standardizovaná protokol vyvinutý speciálně pro komunikaci zařízení mezi systémy automatizace budov, je založen na objektové reprezentaci a organizaci fyzických vstupů a výstupů, stejně jako na nefyzickém konceptu a tím je software. Jde o první mezinárodní standard ISO 16484-5 (od roku 2003) [11]. Cílem byla integrace zařízení od různých výrobců. Každý objekt sítě splňuje

definované požadavky, čímž umožňuje spoluvytvářet síť mnoha propojených objektů nezávisle na zařízení. Protokol lze rozdělit do tří hlavních částí:

2.5.2.1 Objekty

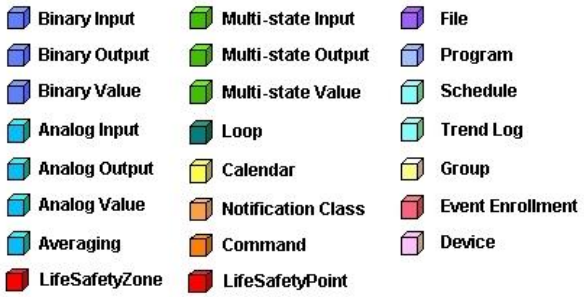
Jakékoliv zařízení, jednotka nebo snímač v síti BACnet je označováno jedním nebo skupinou objektů. Každý objekt je dále charakterizován nastavením vlastností popisující jeho provoz.

Objekt si lze představit jako 2D tabulku, kde nalevo jsou jména informačních položek (název proměnné apod.) a nalevo jejich vlastnosti nebo hodnoty.



Obrázek 5 - Model objektu snímače teploty v BACnetu [11]

Vybrané položky mohou být určeny pouze pro čtení. Ale může jít např. o teplotní čidlo, ze kterého čteme pouze výstup a na jeho základě pracujeme s jinými funkcemi systému. Vstupní hodnota do objektu musí být analogová a tak je nutné nastavit objekt jako Analog input. BACnet definuje kolekci 23 různých standardních objektů:



Obrázek 6 - Přehled základních objektů v BACnetu [11]

Typické BACnet zařízení může obsahovat 16 objektů binárních vstupů a výstupů (BI / BO objects), 2 až 3 objekty plánování apod.

2.5.2.2 Služby

Jedná se o příkazy obsahují informace a úlohy, které se mohou u zvolených zařízení aplikovat. Patří sem služby jako alarmy, časovače, události a trendy.

2.5.2.3 Komunikace

Jedná se o předávání zpráv mezi jednotlivými objekty. Využívají standardní typy médií jako Ethernet, LonTalk, RS232 a RS485. Přímo pro protokol BACnet vznikl speciální typ komunikace BACnet/IP, který kromě posílání zpráv v síti přizpůsobuje jiné technologie [11].

2.5.3 Sběrnice KNX/EIB

Vzniklý protokol EIB v roce 1989 společností EIBA měl za cíl sjednotit využívané přístroje a programovací prostředky. Roku 2001 byl vylepšen protokol společností KONNEX a od té doby nese označení KNX/EIB, který je normalizován normou ISO/IEC 14543 [10].

Protokol patří mezi decentralizované systémy. Umožňuje připojit přídavná zařízení, ale každé z nich musí mít svůj vlastní mikroprocesor a musí být se sběrnicí KNX/EIB kompatibilní. Protokol ke komunikaci využívá tzv. telegramů, které posílá po dvouvodičové datové sběrnici. Telegram si lze představit jako sled binárních hodnot v hexadecimálním kódování. Do sítě je vysílán ve tvaru, kdy počátek řetězce obsahuje start bit. Poté následuje svazek bitů charakterizující příjemce a odesílatele. Po tomto svazku bitů jsou zařazena přenášená data a celý svazek je poté ukončen stop bitem.

Maximální rychlost připojení činní 32 kb/s, v závislosti na použitém médiu a maximální velikost sítě je 1 km. Počet fyzicky spojených zařízení v jedné síti je maximálně 256. Dále sběrnice umožňuje napájení jednotek po sběrnici.

2.5.4 Sběrnice M-BUS

Protokol je určen pro aplikace sběru dat z měřičů a senzorů (např. pitné vody, plynu, tepla). Pracuje na principu *master-slave*, centrální *master* jednotka řídí všechny *slave* jednotky v síti [8].

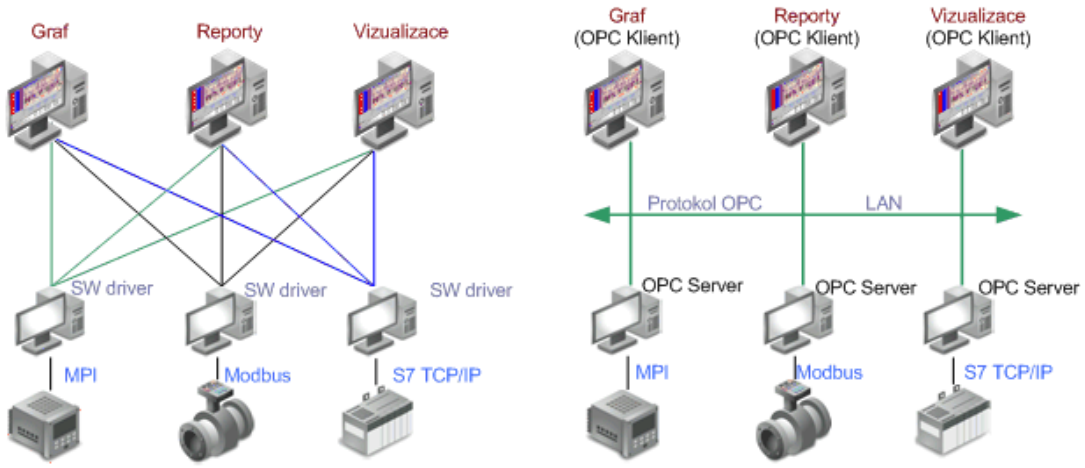
M-BUS využívá pro připojení dvoudrátovou sběrnici, tzn. Snadnou instalaci zařízení. Výrazná výhoda je v tom, že jedna samotná *master* stanice dokáže všem *slave* najednou odesílat data, avšak *slave* stanice neumožní najednou odpovědět. Data může zaslat pouze jedna ze stanic.

Maximální počet zapojených jednotek v síti je 250 (bez implementace síťové vrstvy) s celkovou maximální délkou sítě 1 000 m. Pro maximální dosažitelnou rychlost 9 600 Bd je nutné celkovou délku sítě omezit pouze na 350 metrů.

2.5.5 Protokol OPC

Jedná se o mechanismus pro komunikaci a výměnu dat mezi jednotlivými zařízeními. Data jsou uložena v databázi (OPC server), většinou PC a na žádost jsou zasílána dalším databázím (OPC klient). Podmínkou je existence OPC rozhraní pro obě strany [9].

Přenos dat bez OPC byl problémový z hlediska, že každé zařízení vyžadovalo svůj vlastní ovladač a při instalaci více hardwarových jednotek docházelo k nekompatibilitě apod. Tento problém s OPC odpadá, jelikož jediné rozhraní určené pro komunikaci je právě OPC. Společná komunikační port je Ethernet apod. Do této sítě můžeme bezproblémově přidávat další zdroje signálu (OPC servery) nebo klientské stanice (OPC klienti).



Obrázek 7 - Přenos dat bez OPC (vlevo) a přenos dat pomocí OPC (vpravo) [21]

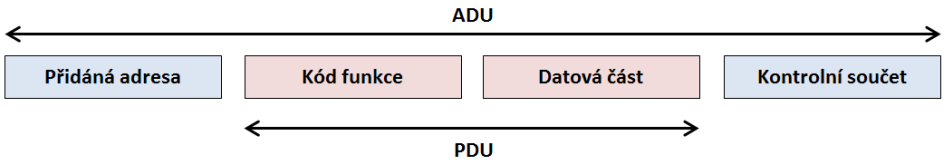
OPC klient i server může být nainstalován na jednom zařízení a může zpracovávat data z více OPC serverů [21].

3 KOMUNIKAČNÍ PROTOKOL MODBUS

Jedná se o komunikační protokol vytvořen v roce 1979 firmou MODICON. Používá se pro vzájemnou komunikaci mezi různými zařízeními na různých typech sítí a sběrnic. Protokol připojí zařízení do stejné sítě (PLC, I/O rozhraní, periferie). Aplikační vrstvu definuje síťový model ISO/OSI[12]. Komunikace probíhá metodou klient-server (*master-slave*), tedy zařízení „*master*“ zasílá dotazy prvku *slave* a ty odpovídají pomocí speciálních kódů.

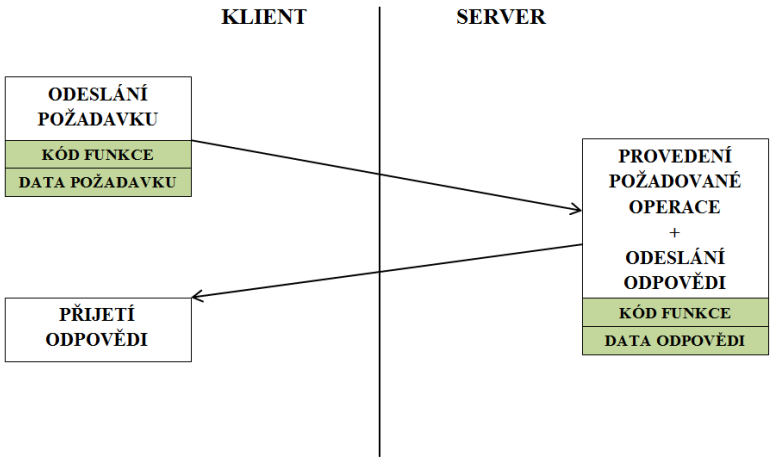
3.1.1 Popis komunikačního protokolu MODBUS

Protokol MODBUS definuje strukturu zprávy na úrovni protokolu (PDU – Protocol Data Unit) nezávisle na typu komunikační vrstvy. PDU dále lze rozšířit v závislosti na typu sítě o další části a tvoří tak aplikační úroveň. Kód funkce udává serveru jaký druh operace má provést. Kód má rozsah 1-255 bitů, přičemž 128-255 bitů (polovina) jsou vyhrazeny pro chybové odpovědi [12]. Druhá polovina kódu obsahuje adresu, počet vstupů, které má server číst nebo hodnota registrů, které zapsat.



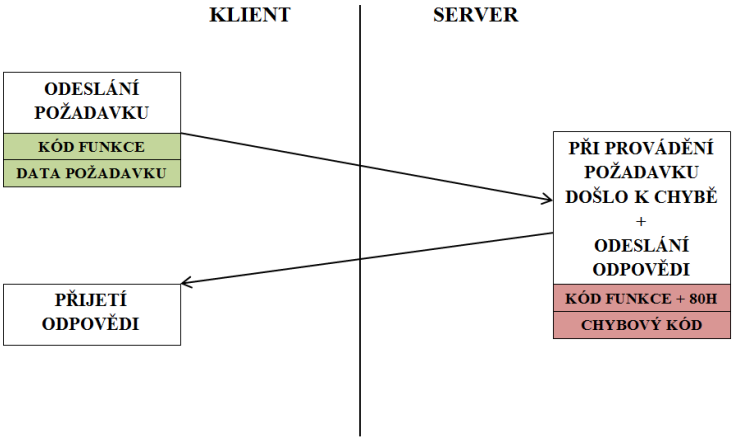
Obrázek 8 - Základní tvar MODBUS zprávy

Pokud při provádění požadavku nedojde k chybě, odpoví server zprávou, která v poli KÓD FUNKCE obsahuje kód operace, kterou požadujeme. V datové části odpovědi předá server klientovi požadavek (obrázek 9).



Obrázek 9 - MODBUS provedení bezchybného požadavku

Pokud při vykonávání operace dojde k chybě, v poli KÓD FUNKCE je stejně jako v předchozím případě vrácen kód operace, avšak s nastaveným nejvyšším bitem, který alarmuje chybu. V datové části je vrácen chybový kód, který poskytuje detailnější popis chyby.



Obrázek 10 - MODBUS provedení chybného požadavku

Jelikož při komunikaci může nastat stav, kdy je možná ztráta požadavku nebo odpovědi. Proto je nutné na straně klienta využít časový limit pro přijetí odpovědi, aby klient nečekal na požadavek, který nikdy nepřijde.
Při první implementaci MODBUSu na sériové lince RS-485 byla zděděna maximální velikost ADU a to 256 bytů[17]. Z toho vyplývá, že maximální velikost PDU činí 253 bytů (jelikož 1 byte je adresa serveru a kontrolní součet CRC činí 2 byty). Podle přechozích obrázků definuje MODBUS tři základní typy zpráv tykající se PDU:
Požadavek (*Request*)

- 1 byte kód funkce, n bytů datová část **požadavku** (adresa, proměnné)

Odpověď (*Response*)

- 1 byte kód funkce (kopie požadavku), n bytů datová část **odpovědi** (přečtené vstupy)

Záporná odpověď (*Exception Response*)

- 1 byte kód funkce + 80h (upozornění na chybu), 1 byte **Chybový kód** (identifikace chyby)

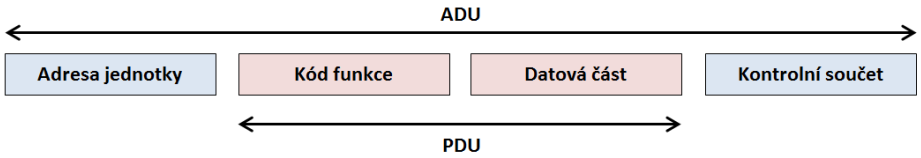
3.1.2 MODBUS na sériové lince (RS-232, RS-485)

V jeden okamžik může být na sběrnici pouze jeden *master* a až maximálně 247 *slave* jednotek. Komunikaci inicializuje vždy *master*, *slave* není oprávněn odesílat odpověď bez povolení *mastera*. *Master* může pracovat ve dvou pracovních režimech: [16], [17]

- unicast – *master* adresuje požadavek konkrétní *slave* jednotce a ta na požadavek odpoví
- broadcast – master pošle požadavek všem *slave* jednotkám, avšak bez zpětné odpovědi

Master nemá přiřazenou specifickou adresu, avšak *slave* jednotky jí mít musí a musí být v celé síti MODBUS jedinečná.

Na obrázku 11 je znázorněn základní formát MODBUS zprávy na sériové lince. Kromě PDU obsahuje zpráva adresu jednotky, která je právě jedinečná v celé síti a odkazuje na *slave* jednotku. Kontrolní součet slouží k detekci chybné odpovědi a obsahuje CRC a LRC kód.

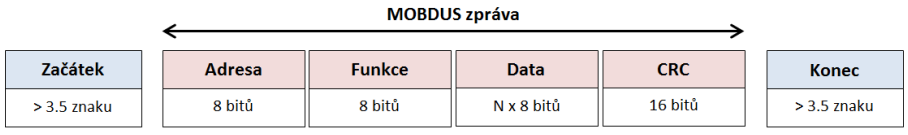


Obrázek 11 - MODBUS zpráva na sériové lince

MODBUS definuje na sériové lince dva vysílací režimy, MODBUS RTU a ASCII. Režim určuje v jakém formátu jsou data vysílána a dekódována, režim RTU je velice rozšířen a v dnešní době se počítá s přítomností u všech PLC, zatímco ASCII je nepříliš populární tak jsou požadavky na něj méně vyžadovány a ne každé PLC ho podporuje. Požadavek je, že všechny jednotky na sběrnici využívají stejný vysílací režim.

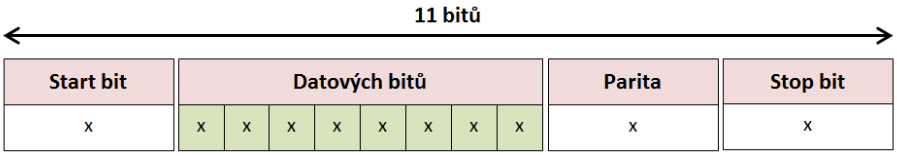
3.1.3 MODBUS RTU

Režim RTU obsahuje každý 8-bitový byte zprávy dva 4-bitové hexadecimální znaky. Vysílané zprávy musí být souvislé, mezera mezi znaky nesmí přesáhnout velikost 1.5 znaku. Začátek a konec zprávy je identifikován prostorem na sběrnici delší než 3.5 znaku. K detekci chyb slouží 16-bitové CRC pole.



Obrázek 12 - MODBUS zpráva v režimu RTU

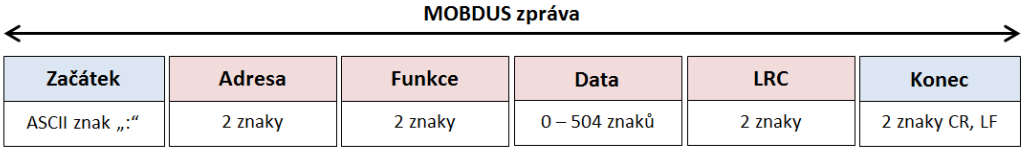
Jednotka musí podporovat sudou paritu, pokud není parita využita, je nahrazena druhým stop bitem.



Obrázek 13 - Formát bytu u režimu MODBUS RTU

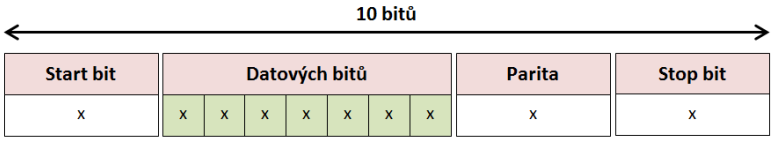
3.1.4 MODBUS ASCII

V tomto režimu je každý 8-bitový byte posílán jako dvojice ASCII znaků. Umožňuje vysílat znaky s mezerami do 1 s. Začátek zprávy je identifikován znakem „:“ a konec zprávy řídicími znaky CR, LF. Oproti režimu RTU je pomalejší a méně rozšířený.



Obrázek 14 - Zpráva v režimu ASCII

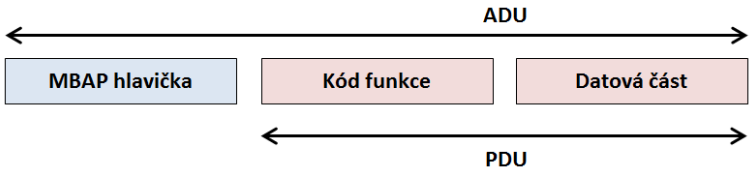
Chybu detekujeme pomocí 8 bitového pole LRC.



Obrázek 15 - Formát bytu u režimu MODBUS ASCII

3.1.5 MODBUS TCP/IP

Pro identifikaci MODBUS ADU je použita MBAP hlavička (MODBUS Application Protocol Header).



Obrázek 16 - Základní tvar MODBUS zprávy na TCP/IP

Pro odesílání MODBUS/TCP ADU je speciálně na TCP vyhrazen port 502[12]. Využívá klasický Ethernet TCP/IP s rychlostí 10/100 Mbit/s (postačuje standardní síťová karta). Princip komunikace *1Master* x *nSlave* [3]. Velikost ADU na TCP/IP = 253 bytů PDU + MBAP = 260 bytů. Pro datový model MODBUS jsou definovány 4 základní tabulky:

Tabulka 1 – Datový model MODBUS [12]

Název	Typ položky	Přístup	Mapování (S7-1200)	Adresa
Diskrétní vstupy (Discrete Inputs)	1 bit	READ	I0.0 – I1023.7	1-8192
Cívky – Výstupy (Coils) – Outputs	1 bit	READ WRITE	Q0.0 – Q1023.7	1-8192
Vstupní registr (Input Registers)	16 bitové slovo (1 WORD)	READ	MB_HOLD_REG	
Holding registr (Holding Registers)	16 bitové slovo (1 WORD)	READ WRITE	MB_HOLD_REG	

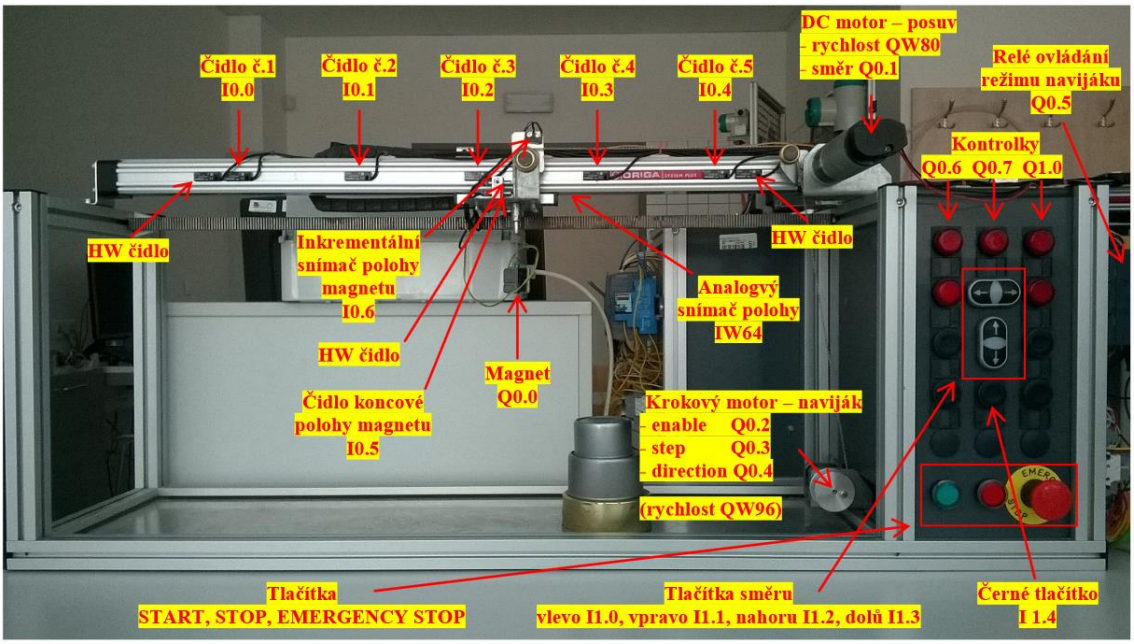
4 PROPOJENÍ HAWK S REÁLNÝM MODELEM PŘES MODBUS TCP

Pro demonstrování funkčnosti komunikace mezi reálným modelem a HAWK regulátorem pomocí MODBUS TCP/IP byl vybrán laboratorní model portálového jeřábu, který je k dispozici v laboratoři moderních metod v automatizaci.

4.1 Popis PLC modelu

Pro demonstrační úlohu funkčnosti komunikace byl použit model jednoho z jeřábů. Model je vybaven pěti funkčními tlačítky a třemi funkčními kontrolkami. Jeřáb disponuje hlavním ramenem (osa x). Rameno je vybaveno pěti senzory polohy a dvěma bezpečnostními snímači na jeho koncích. Posuvnou částí po hlavním rameni je navíjející elektromagnet (osa y). Model dále obsahuje ve spodní části tlačítko EMERGENCY STOP, které hardwarově vypne chod motorů. Při aktivaci bezpečnostního senzoru se tlačítko STOP na obrázku 17 rozsvítí a model přestane reagovat na příkazy do doby, než model opět aktivujeme stisknutím tlačítka START.

Programovatelný automat použitý jako model pro sběr dat je od firmy Siemens s označením S7-1200 a je rozšířen o modul analogového výstupu SB1232 – AQ 1x12BIT a jednu I/O analogovou kartu SM 1234 – AI 4x13BIT / AQ 2x14BIT [24].



Obrázek 17 - PLC model využitý při demonstraci MODBSU TCPCP/IP komunikace a realizaci praktické úlohy [24]

4.1.1 Ovládání pohonů

Pro horizontální posuv hlavního jeřábu (osa x) je k dispozici stejnosměrný motor M24x40. Motor je ovládán dvěma proměnnými. Jednou z nich je napětí (v automatu realizováno pomocí numerické hodnoty v rozsahu od 0-15000), které přivádíme na výstupní adresu QW80. Přičemž při zapsání hodnoty 0 se motor zastaví. Druhou proměnou je směr motoru, který je řízen pomocí digitálního výstupu (log. 0 = směr vlevo z pohledu obsluhy).

Navíjení magnetu ovládáme pomocí motoru Microcon SX23-1020, jde o dvojfázový krokový motor se základním úhlem 1,8° (úhel na jeden krok) [24]. Motor přes prostředí TIA PORTAL ovládáme pomocí tzv. *Technology object AXIS*, přičemž nakonfigurujeme základní parametry motoru a s motorem později pracujeme jako s jedním blokem (v LAD). Jde o jednodušší rešení, než volit ovládání motoru přes analogové ovládání, které aktivujeme přivedením logické 1 na digitální výstup Q0.5.

4.1.2 Prostředí TIA PORTAL

Použitá verze prostředí k programování vlastního zdrojového kódu pro průmyslové automaty je verze 13 (označováno jako V13) licencovaná pro laboratoř předmětu BPPA. Prostředí představuje společné vývojové prostředí pro přípravu, realizaci a údržbu automatizačních projektů vyvíjené firmou Siemens. TIA nabízí řešení s jednotnou HW i SW základnou pro různé druhy

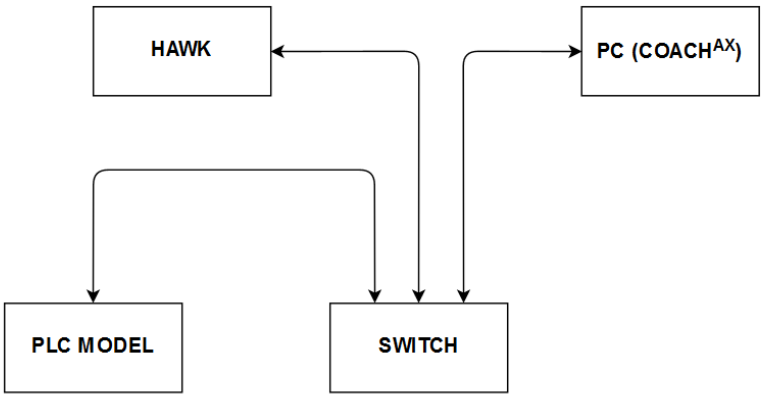
automatizačních úloh ve veškerých automatizačních oborech např. HW konfigurace, sledování stavů a debugging. Velkou výhodou prostředí je přehlednost a flexibilita [14].

Zdrojové kódy pro praktickou část úlohy byly vytvářeny v programu Simatic Step7 V13 a možnost nahlédnout do jejich kódu je v elektronické příloze BP_PRILOHA_PLC.

4.2 Popis testovací úlohy

Pro demonstraci funkčnosti byla zvolena jednoduchá změna decimální hodnoty, která prezentuje pomocnou proměnnou v PLC programu, přenášenou pomocí holding registru a binární hodnoty (tzv. *coil*, jelikož se jedná o digitální výstup) skrze prostředí COACH^{AX}.

Programovatelný automat je v pozici MODBUS server, jelikož vytváříme nadřazený systém, který bude sbírat data z různých modelů, tzn. server bude zpřístupňovat své vstupy/výstupy a holding registry nadřazenému zařízení (HAWK). Regulátor v pozici klienta se bude dotazovat na potřebné stavy a hodnoty proměnných v PLC modelu (serveru).



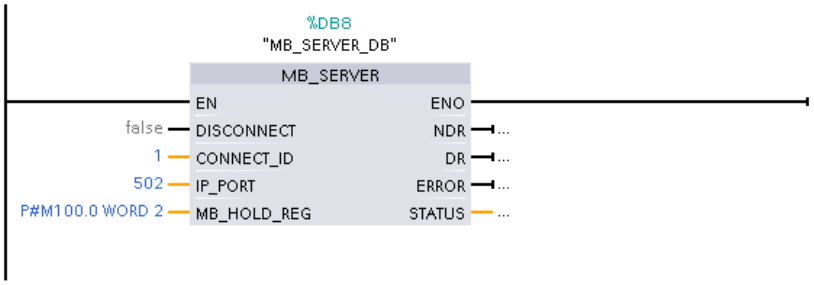
Obrázek 18 - Blokové schéma zapojení při demonstraci komunikaci přes MODBUS TCP/IP

4.3 MODBUS konfigurace PLC modelu

V hlavním bloku automatu (OB1) jsme vytvořili funkční blok s názvem *SOURCE*, který obsahuje zdrojový kód řízení automatu a v OB1 jej voláme. Druhou větev v hlavním bloku použijeme pro založení MODBUS SEVERU. Tuto hierarchii volím pro lepší přehlednost. PLC bude v pozici serveru odesílat dat do HAWKu. Prvním krokem, který je nutný podstoupit, je přesunout automat do stejné podsítě. Původně byla adresa automatu 192.0.1.10, avšak HAWK byl v jiné podsíti, ověřeno

pomocí příkazového řádku na počítači s nainstalovaným prostředím COACH^{AX} a připojeným integrátorem (cmd příkaz *ipconfig /all*).

V hardwarové konfiguraci PLC jsme změnili jeho IP adresu na stejnou podsíť jako integrátor. V tomto případě je volba změny IP u PLC lepší, než u integrátoru z důvodu jednoduššího uživatelského prostředí ze strany SIEMENSu a celkové konfigurace, jelikož HAWK se používá i pro jiné úlohy a pevně měnit jeho IP je zbytečná komplikace.



Obrázek 19 - Nastavení PLC modelu jako MODBUS server v prostředí TIA PORTAL V13

Blokem na obrázku 19 nastavíme PLC jako MODBUS server. Vstupy *CONNECT_ID* značí označení zařízení v síti, *IP_PORT* port připojení (výchozí 502 ponecháme, protože je to vyhrazený port pro MODBUS TCP/IP[12]) a *MB_HOLD_REG* značí vyhrazené místo pro holding registry, pomocí kterých si budeme posílat *read-write* decimální hodnoty (Podle MODBUS tabulky se jedná o holding registry). Velikost odesílaného požadavku pomocí HR je od MW100 velikost 2x WORD (4 byty) [25].

Po levé straně, v konfiguraci automatu najdeme položku *TAGS*, jedná se o proměnné a zde vytvoříme jednu pomocnou proměnnou na adrese MW100, a binární výstup na adrese Q0.0, nesmíme zapomenout na volbu typu proměnné.

Tímto řešíme veškerou konfiguraci na straně PLC, důležité je zapojit model do sítě k HAWK pomocí switche a UTP kabelu.

Zdrojový kód pro průmyslový automat je naprogramován v jazyce LAD z důvodu lepší přehlednosti a tvoření algoritmů je srozumitelnější.

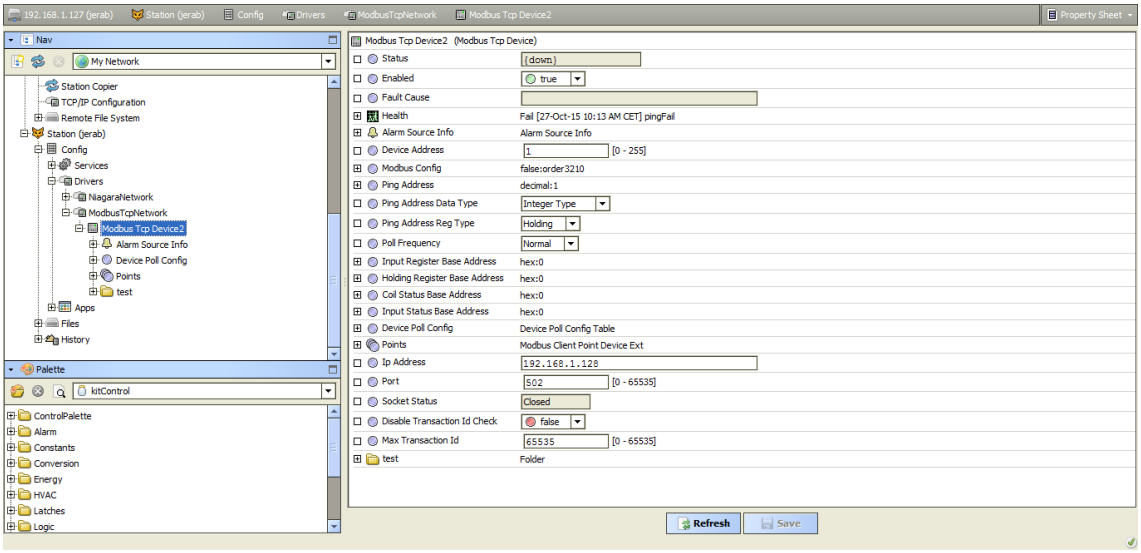
4.4 MODBUS konfigurace na HAWK regulátoru skrze COACH^{AX}

Prvním krokem k vytvoření klienta MODBUS je otevřít novou platformu, kterou bude samotný HAWK. K integrátoru se připojíme pomocí jeho IP adresy (v přiloženém dokumentu nebo jej zjistíme pomocí příkazu *ipconfig /all*). Přihlašovací údaje jsou: přihlašovací jméno: tridium, heslo: niagara1.

Poté vytvoříme stanici přímo v nové platformě (HAWK), kam zvolíme vlastní přihlašovací údaje a heslo kliknutím pravým tlačítkem myši na platformu ->

NEW -> Station. V prostředí může být aktivní vždy pouze jedna stanice. Proto kontrolu stanice, provedeme v možnostech platformy a aplikace *Application manager*. Naše nová stanice má mít status *Running* nebo *Starting* (tento stav nás informuje o spouštění aplikace a nelze se na ní připojit, musíte vyčkat). Pokud tomu tak není a je aktivní stanice s jiným názvem, provedeme STOP a následný start námi vytvořené nové stanice.

Ve stanici musíme vytvořit v ovladačích (*Drivers*) novou MODBUS síť. Tu přidáme do palety pomocí knihovny *ModbusTcp* položku *ModbusTcpNetwork*. Jelikož HAWK bude v MODBUS síti klientem, je nutné přidat do sítě zařízení reprezentující programovatelný automat. V podmenu knihovny *ModbusTcp* vybereme položku *ModbusTcpDevice*. Přetažením zařízení na stanici jej přidáme a zadáme IP adresu automatu. *Device Address* volíme stejně jako *CONNECT_ID* při konfiguraci MODBUS serveru.



Obrázek 20 - Konfigurace server MODBUS zařízení v prostředí COACH^{AX}

Po vyplnění údajů rozevřeme přehled sítě (*ModbusTcpNetwork*) a pravým tlačítkem klikneme na libovolné místo řádku se zařízením, kde zvolíme možnost Action-> Ping. Při úspěšné inicializaci MODBUS klienta se na pravé straně řádky objeví hláška *Opened*. Tím jsme úspěšně spojeni se serverem a můžeme začít s požadavky pro sběr hodnot.

4.4.1 Použitý HAWK integrátor a prostředí COACH^{AX}

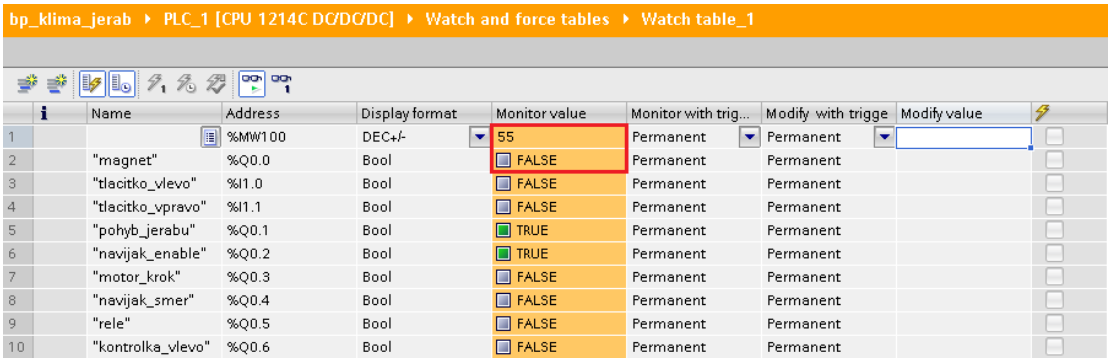
Při vytváření praktické části úlohy bylo k dispozici vývojové prostředí COACH^{AX} verze 3.8.38.2 a HAWK integrátor verze 300E, který pracuje na architektuře ppc. Obsahuje jedno-jádrový procesor s taktem 524 MHz, 260 MB RAM a uložiště pro projekt o velikosti přibližně 128 MB [2]. Je vybaven dvěma ethernetovými porty,

sériovými porty pro RS232 a RS485. K regulátoru je připojen externí modul CLAXHAWKIO34, který obsahuje 16 univerzálních vstupů, 20 digitálních a 8 analogových výstupů [2].

4.5 Demonstrace funkčnosti na změně stavu výstupu

V prostředí COACH^{AX}, konkrétně v *ModbusTcpDevice* (zařízení) jsme vytvořili decimální a binární hodnotu. Jelikož se jedná v protokolu MODBUS o model *coil* a holding register. Tento typ tabulky je *read-write*, teda přepisovatelné (možnost zápisu i čtení), pokud bychom chtěl číst stav binárních vstupů (*Input*) budeme muset zvolit *Constant Point* a v jeho nastavení přepnout *Status type* na *Input*, pak podle decimálního čísla zvolíme adresu bitu 0 prezentuje I0.0 a 1 decimální, prezentuje I0.1. V nastavení jednotlivých proměnných jsme nastavili nejnižší možnou odezvu na změnu hodnoty, jelikož v pozdější části práce se stávalo, že vizualizace zobrazila aktualizaci hodnoty těsně před její další změnou stavu, např. změna směru jeřábu před zastavením na snímači.

V prostředí TIA PORTAL si zvolíme tabulku proměnných (*Watch table*), abychom mohli sledovat stav vstupů i výstupů. V této demonstrující úloze sledujeme pouze výstup magnetu a decimální hodnotu pomocné proměnné. Červeně označeným hodnotám na řádcích číslo 1 a 2 budeme měnit stav.



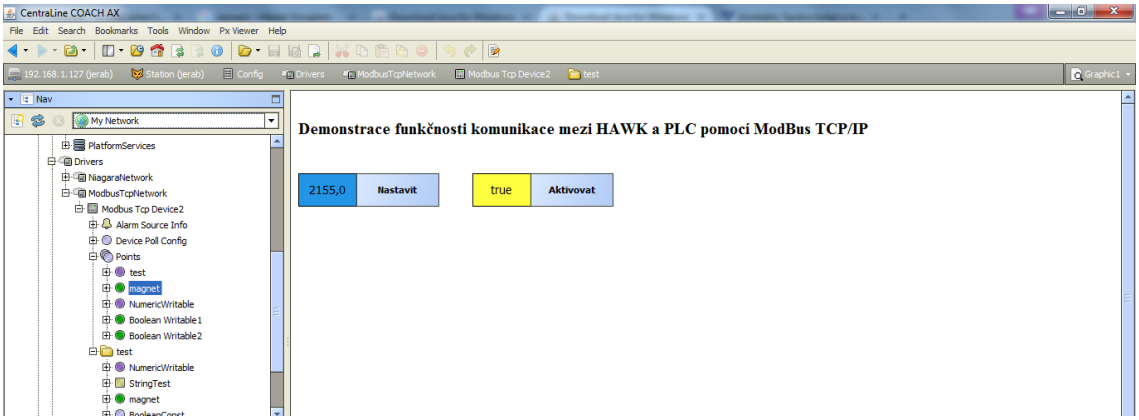
	Name	Address	Display format	Monitor value	Monitor with trig...	Modify with trigge	Modify value
1		%MW100	DEC+/-	55	Permanent	Permanent	
2	"magnet"	%Q0.0	Bool	FALSE	Permanent	Permanent	
3	"tlacitko_vlevo"	%I1.0	Bool	FALSE	Permanent	Permanent	
4	"tlacitko_vpravo"	%I1.1	Bool	FALSE	Permanent	Permanent	
5	"pohyb_jerabu"	%Q0.1	Bool	TRUE	Permanent	Permanent	
6	"navijak_enable"	%Q0.2	Bool	TRUE	Permanent	Permanent	
7	"motor_krok"	%Q0.3	Bool	FALSE	Permanent	Permanent	
8	"navijak_smer"	%Q0.4	Bool	FALSE	Permanent	Permanent	
9	"rele"	%Q0.5	Bool	FALSE	Permanent	Permanent	
10	"kontrolka_vlevo"	%Q0.6	Bool	FALSE	Permanent	Permanent	

Obrázek 21 - Hodnoty a stavy proměnných před změnou v ovládacím prostředí COACH^{AX}

V dalším obrázku bude k dispozici náhled do prostředí, z kterého se budou měnit hodnoty. Jejich aktuální hodnoty zobrazíme v grafickém prostředí pomocí pouhého přetažení datového typu na plochu, v zobrazeném podmenu máme na výběr zobrazení pouze hodnoty a to označením *Bound label*, dále blok pro změnu nebo např. grafickou závislost v čase.

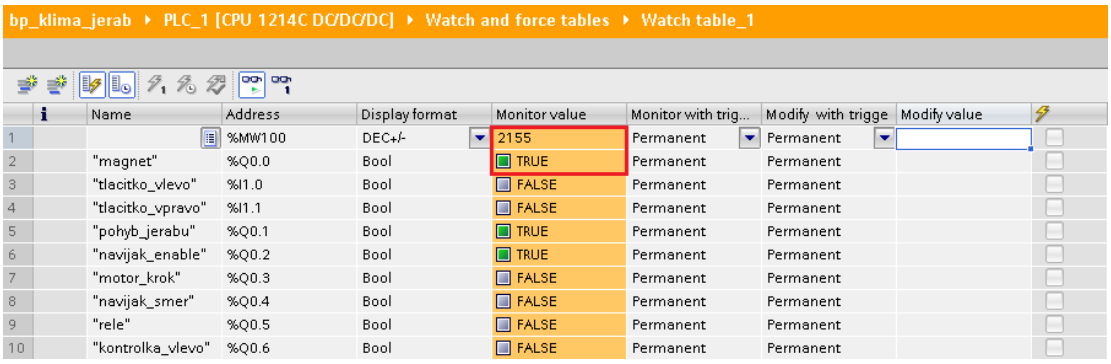
Na pravé straně v nastavení MODBUS zařízení, rozklikneme položku *Points*. V této položce vybereme nový *point* (*New*) a pro změnu binárního výstupu vybereme položku *BooleanWriteable*, jelikož se jedná podle MODBUS tabulky o *coil* (tedy také možnost zápisu, nejen čtení) a podle adresy na PLC jej zvolíme

případnou decimální adresu. Podle dokumentace modelu automatu doložené k laboratorní úloze předmětu BPPA se nachází magnet na adrese Q0.0, této adrese v decimální hodnotě odpovídá 0. Právě zvolením *BooleanWriteable* automaticky prostředí bere, že budeme číst výstupní hodnoty automatu, v případě čtení pouze vstupní hodnoty (tedy IX.X) se budeme věnovat v pozdější části práce. Vše potvrdíme tlačítkem OK. Numerickou hodnotu přidáme pomocí *NumericWriteable*, jelikož budeme hodnotu měnit. Dle datové tabulky MODBUS musíme využít typu holding registrů, které prostředí automaticky bere po zvolení položky *NumericWriteable*. Přenos pomocí HR máme adresovaný na straně serveru a to od adresy M100.0 což odpovídá adrese MW100. Decimální adresa v prostředí COACH^{AX} pro MW100 činí 0. Jelikož na této adrese je vyhrazena paměť pro HR. Jednoduchým přetažením pointů na pracovní plochu provedeme lehkou vizualizaci a čtení stavů nebo hodnot proměnných. Na obrázku 22 můžete vidět nastavenou decimální hodnotu a změnu binárního výstupu.



Obrázek 22 - Vizualizace ovládacího panelu pro změnu výstupní a pomocné hodnoty v PLC

Na obrázku 23 je možné pozorovat důkazné změny v prostředí TIA PORTAL V13 se změněnými hodnotami.



	Name	Address	Display format	Monitor value	Monitor with trig...	Modify with trigge	Modify value
1		%MW100	DEC+/-	2155	Permanent	Permanent	
2	"magnet"	%Q0.0	Bool	TRUE	Permanent	Permanent	
3	"tlacitko_vlevo"	%I1.0	Bool	FALSE	Permanent	Permanent	
4	"tlacitko_vpravo"	%I1.1	Bool	FALSE	Permanent	Permanent	
5	"pohyb_jerabu"	%Q0.1	Bool	TRUE	Permanent	Permanent	
6	"navijak_enable"	%Q0.2	Bool	TRUE	Permanent	Permanent	
7	"motor_krok"	%Q0.3	Bool	FALSE	Permanent	Permanent	
8	"navijak_smer"	%Q0.4	Bool	FALSE	Permanent	Permanent	
9	"rele"	%Q0.5	Bool	FALSE	Permanent	Permanent	
10	"kontrolka_vlevo"	%Q0.6	Bool	FALSE	Permanent	Permanent	

Obrázek 23 - Hodnoty a stavy proměnných po změně v ovládacím prostředí COACH^{AX}

Stejným způsobem budeme v praktické úloze zpracovávat data z celého modelu a vizualizovat jeho chování. Důležitým faktorem pro sestavení kompletní vizualizace je inicializace všech žádaných proměnných. V praktické úloze chci aplikovat částečné řízení modelu v prostředí COACH^{AX} (z pohledu obsluhy u PC).

5 NÁVRH A REALIZACE PRAKTICKÉ ÚLOHY PRO HAWK REGULÁTOR

Praktickou úlohou se rozumí vytvoření nadřazeného systému pro sběr dat, který byl aplikován na modely portálového jeřábu, popsané v předchozí kapitole, ke kterým byl vytvořen rozdílný zdrojový kód v prostředí TIA PORTAL. Zdrojové kódy se od sebe liší ovládací metodou. Zatímco jeden z modelů je plně manuální, tím se rozumí ovládání obou os (x a y) pouze volbou obsluhy, druhý model je plně automatizovaný a přenáší objekt z jednoho konce pracovní plochy jeřábu na druhou. Z těchto dvou modelů jsou data přenášena pomocí komunikačního protokolu MODBUS na TCP/IP nadřazenému systému, kterým je regulátor HAWK v pozici klienta v MODBUS síti. Ze sbíraných dat ve vývojovém prostředí COACH^{AX} bude vytvořena vizualizovaná aplikace (obrázek 28). Pro účely této aplikace je nezbytná konfigurace sítě, vytvoření řídicí logiky, historie, statistik, ovládání a monitoringu. Práce navazuje na předchozí kapitolu, kde se pomocí vývojového prostředí měnil výstup na modelu jeřábu.

První model je ovládán pomocí směrových tlačítek jednoduše ovládá pohyb jeřábu. Model je vybaven pěti snímači pro zjištění polohy osy x. Po stisknutí jednoho z tlačítek popojede jeřáb na další snímač. Objekty, se kterými model manipuluje, můžeme položit na pozici snímačů, což přináší jednoduší manipulaci při testování modelu.

Jedním z problémů při tvoření programu byla rychlost motoru, jelikož bylo aplikováno automatické navíjení navijáku (v ose y), kde se při větší rychlosti nestihlo navinout lanko a v důsledku toho se sepnul bezpečnostní snímač na navijáku, který zastavil celý model, byla tedy zvolena vhodná rychlost, která splňuje všechny nároky na bezpečný chod modelu.

Druhý model je téměř celý automatický. Z jednoho konce pracovní plochy přenese objekt na druhý konec. Jedním z problému bylo namotávání lanka, které je při pohybu ramena důležité. Protože může docházet k zamotávání nebo přebytečnému odmotávání lanka, vedoucí k zaseknutí motacího bubnu. Další překážkou byla identifikace země, tedy pokud magnet dosáhl objektu pro zvednutí a tím pádem měl být zastaven navíjecí režim modelu. Tento problém se řešil pomocí inkrementálního snímače na pohyblivé části magnetu (po ramenu). Pokud

se stav inkrementálního snímače po dobu jedné vteřiny neměnil, program identifikuje dojetí magnetu k objektu.

Celý proces se spouští jedním funkčním tlačítkem, které musí být ošetřené proti počátečnímu zamotávání lanka tím, že po stisknutí tlačítka se konstantně odvine lanko o přibližně 5 cm, jelikož vznikal havarijní stav kvůli přebytečnému namotání lanka a spínal se bezpečnostní snímač na konci lanka.

Jelikož jsou modely v postavení serveru, dotazuje se HAWK regulátor na stavy a hodnoty přenášené pomocí jednotlivých bitů nebo pomocí tzv. holding registrů.

Tabulka 2 - Seznam činností vytvořeného nadřazeného systému

Název funkce	**Nutnost programování**	**Zásahu do PLC prog.**	**Ovlivnění chodu modelu**
Identifikace sepnutí magnetu	NE	NE	ANO
Okamžitá hodnota rychlosti hlavního motoru	ANO	ANO	ANO
Režim navijáku	NE	NE	NE
Bezpečností okruh	NE	NE	ANO
Směr a identifikace pohonu jeřábu	NE	ANO	NE
Počet převezených objektů + počet přejetých senzorů	ANO	ANO	NE
Aktuální poloha jeřábu + vykonávaná činnost	ANO	NE	NE
Aktuální účinnost zařízení (v %)	ANO	NE	NE
Nulování čítačů	NE	ANO	NE
Nastavitelná rychlost hlavního motoru pomocí posuvníku	ANO	ANO	ANO
Kompletní vizualizace obou zařízení	ANO	ANO	ANO
LOG + informace o komunikaci	ANO	NE	NE
Upozornění na počet aktivovaných senzorů	ANO	ANO	NE
Grafická závislost obou aktuálních rychlostí motorů včetně jejich průměru na čase	NE	NE	NE
Grafická závislost aktivovaných snímačů včetně jejich součtu na čase	NE	NE	NE
Grafická závislost přenesených objektů včetně jejich součtu na čase	NE	NE	NE
Možnost ovládání modelu skrze prostředí COACH^{AX} a možnost ovlivnit chování modelu s téměř okamžitou odezvou	ANO	ANO	ANO

Oba modely jsou vybavené čítači pro počet přenesených předmětů (ošetření ze strany operátora a obsluhy PC je aplikováno tak, že pokud je magnet aktivní, čítač není aktivní).

5.1 Konfigurace sítě

Při konfiguraci sítě pro realizaci praktické úlohy volíme stejným způsobem jako v kapitole 4. S tím rozdílem, že v MODBUS síti budou dvě zařízení. Konfigurace se liší v zadání IP adresy a decimální adresy v síti každého z modelů.

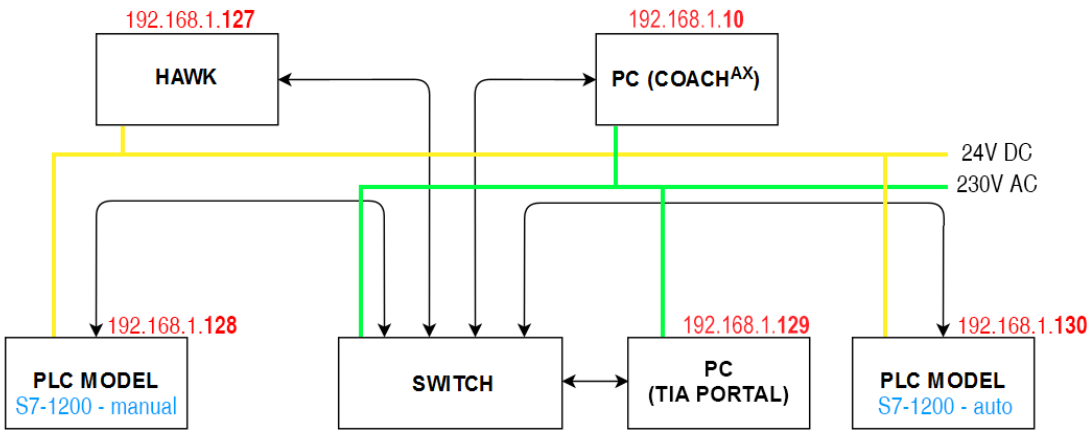
Síť vytvoříme přetáhnutím položky *ModbusTcpNetwork* z knihovny *ModbusTcp* do pracovního stromu v nastavení stanice. Přidáním položek *ModbusTcpDevice* přidáváme konkrétní reálné zařízení (PLC), kde nastavíme, jak bylo zmiňováno v předchozí kapitole, tedy každá nová položka *Device* prezentuje náš reálný model, který se v konfiguraci liší IP adresou a *Device Address*, která se musí shodovat se vstupem *CONNECT_ID* v konfiguraci na straně server, tedy PLC. Viz obrázek 19.

Database

Name	Exts	Status	Device Address	Ip Address	Port	Socket Status
Modbus Tcp Device2		{ok}	1	192.168.1.128	502	Opened
model_2		{ok}	2	192.168.1.130	502	Opened

Obrázek 24 - Náhled do MODBUS sítě při realizaci praktické úlohy

Na obrázku 24 je zobrazena MODBUS síť včetně statusu komunikace, který je u obou zařízení *Opened*, tedy otevřen a připraven na komunikaci, pokud by se objevil status *Closed*, který se aktivuje po delší nečinnosti, klikneme pravým tlačítkem na zařízení a zvolíme položku *Actions -> Ping*. Zařízení opět otevře komunikaci. Síť nekonfigurujeme, pouze vytvoříme přesunutím, jak bylo zmiňováno výše. Zařízení je nutné konfigurovat, popis procesu je popsán v kapitole 3. Jak již bylo zmiňováno, důležitým faktem je přesunout všechny zařízení spojená s vytvářením úlohy do stejné podsítě.

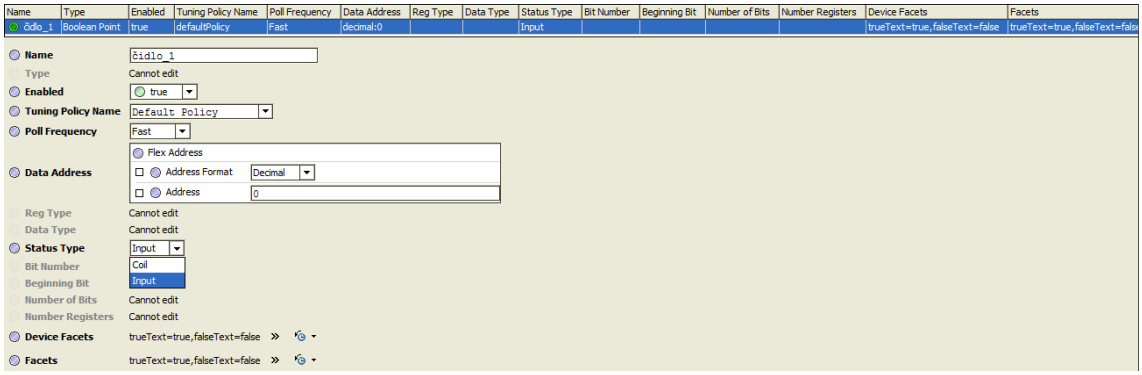


Obrázek 25 – Blokové schéma zapojení pracoviště pro realizaci praktické části úlohy

5.2 Stavy a hodnoty proměnných z PLC a jejich přenesení do prostředí COACH^{AX}

Při úspěšném propojení PLC modelů a prostředím COACH^{AX}, přihlášeného k HAWK regulátoru (platformě) chceme číst stavy a hodnoty proměnných. Budeme se řídit podle základních tabulek MODBUSU (Tabulka 1). V tabulce je možnost si povšimnout velikosti paměti a zápisových práv.

Ve vývojovém prostředí COACH^{AX}, v nastavení zařízení otevřeme položku *Points,* prostředí čte a pracuje se stavy proměnných pomocí tzv. bodů. V tomto okně klikneme pro přidání nového pointu příkazem ve spodní části obrazovky *NEW*. Zde se řídíme podle tabulky 1, pokud je velikost paměti 1 bit, jedná se o datový typ boolean. Zde je nutnost brát zřetel jestli čteme přímo vstupy (značené I) na automatu, které mají právo pouze *READ* nebo výstupy (značené Q), kde můžeme stav číst, ale i měnit. Pro přidání hodnoty vstupu přidáme položku *BooleanConstant* a při nastavování parametrů pointu zvolíme *Status type* na *INPUT.* Tím označíme proměnnou jako vstup a pomocí decimální hodnoty adresy (*Data address*) zvolíme požadovaný bit na automatu. Decimální hodnotě 0 odpovídá vstup I0.0.



Obrázek 26 - Nastavování parametrů bodu (pointu) čidlo_1 typu boolean

Položku *Pool Frequency* volíme co nejvyšší (tedy hodnotu *Fast*), jelikož chceme co nejmenší odezvu při změně stavu. Pokud chceme ovlivňovat nebo číst výstup automatu, zvolíme položku *BooleanWriteable*, jelikož můžeme měnit jejich vztah a prostředí tento bod automaticky bere jako datový model *coil*, tedy položku, která má atributy čtení-zápis a decimální hodnota adresy 0 odpovídá výstupu Q0.0.

Pokud chceme přenášet např. decimální hodnotu, která prezentuje rychlost motoru musíme využít tzv. holding registry. Ten přidáme mezi pointy přidáním *NumericWriteable* položkou. COACH^{AX} ji automaticky pro své práva (čtení-zápis) bere jako holding registr, z toho vyplývá, že nebereme zřetel s nastavováním statusu. Při volbě adresy volíme decimální hodnotu 0, která odpovídá MW100.

Jelikož bylo MODBUS serveru vyhrazené místo od adresy M100.0 (obrázek 19) odpovídající zmiňované adrese MW100.

Je nutné podotknout, že přepisovatelné (*Writeable*) položky je možné měnit přímo v položce *Points*. Bool stavy přepínat (*true/false*) a numerické hodnoty zadávat ručně.

5.2.1 Přenesení dat na pracovní plochu

Při přidání všech potřebných proměnných je důležitá manipulace s nimi. Je důležité podotknout, že každý point může být inicializován pouze jednou, tzn. pokud jej přetáhneme na pracovní plochu a pracujeme s jeho hodnotou, nelze tak učinit v dalším podprogramu. Řešení spočívá v nalinkování vzájemných vztahů, vytvoříme prázdnou proměnnou stejného datového typu a na vstup této proměnné propojíme pomocí *link mark* výstup naší původní proměnné.

Programy tvoříme do tzv. *Folders*, pravým tlačítkem klikneme na zařízení *New -> Folder*. Pokud složku vytvoříme v zařízení, doporučuji ji pouze pro podprogram neboli mimo-výpočet, hlavní aplikaci vytvoříme v *Apps -> New -> Folder*, jelikož chceme sbírat data z více modelů a z pohledu základní hierarchie programu je toto řešení nejpřehlednější. Složka v zařízení slouží například k výpočtu standardizace rychlosti motoru.

V hlavním stromovém adresáři *Apps* při rozkliknutí složky můžeme vytvořit nový pohled a prostředí nám automaticky nabídne grafický pohled (vizualizaci), vše ve výchozím nastavení potvrdíme. Pokud chceme zobrazit stav proměnné, v levém menu podle zařízení najdeme požadovanou položku a pouze přeneseme na pracovní plochu prostředí. Je nutné podotknout, že musíme být v pohledu úpravy grafiky, na který se přepneme pomocí pohledů na pravé straně obrazovky (obrázek 2). Při přetažení položky na plochu se zobrazí okno nastavitelných parametrů pro proměnou, *Bound label* položka zobrazí aktuální hodnotu proměnné. Stav stringů *true/false* na uživatelsky volitelný, lze změnit v nastavení pointu v řádce *Facets,* například na *pravda/nepravda.* U decimálních hodnot lze měnit jednotky nebo například počet čísel za desetinnou čárkou.

5.3 Ovládací prvky v prostředí COACH^{AX}

Jelikož je vytvářen nadřazený systém, měla by být možnost proces ovlivnit na vyšší úrovni, což zahrnuje také kompletní ovládání modelu. Vizualizována byla kopie reálného modelu automatu do prostředí včetně všech tlačítek. Vizualizovaný model reaguje stejným způsobem jako model reálný.

Při vytváření ovládání skrze vizualizaci bylo nutné vytvořit pomocné proměnné typu *Writeable* (přepisovatelné), jelikož tlačítka modelu jsou vstupy a ty nelze samovolně měnit, jelikož jsou typu *Input* a ty lze pouze číst.

V programu automatu byla vytvořena paralelně větev k tomuto ovládacímu prvku, který byl adresován na výstupu automatu a nastaven datový model *coil*. V předchozí kapitole bylo zmiňováno, že je možné měnit stav přímo v prostředí, avšak je požadováno zastávat funkci tlačítka, tedy aktivní stav pouze omezenou krátkou dobu (jednotky desetin vteřiny). Opakujeme podobný postup jako při zobrazování hodnoty proměnné v kapitole 4.3. Přetáhneme pomocnou proměnou na pracovní plochu v režimu *Edit Graphics* a při zadávání parametrů vybereme možnost *Actions*. Nyní v pravé části okna je možné zvolit akci, kterou má objekt zastávat. Pokud stistknutím tlačítka je vyžadováno proměnnou aktivovat, zvolíme možnost *Active*. V editaci položky, ve spodní části obrazovky, změníme stav položky *activePermanent.* Na možnost nastavení času uživatelem a zvolíme čas 500 ms. Tímto je tlačítko aktivní po dobu půl vteřiny a tím simuluje reálné tlačítko.

5.4 Přístup k uživateli vytvořeným souborům

V případě tvoření vizualizace bylo v prostředí nedostatečné množství vyhovujících prvků z dostupných knihoven a to obrázků nebo funkcí. Prostředí nabízí možnost flexibility a tím i implementaci vlastních obrázků nebo knihoven. Například pro vytvoření boolean obrázku, kdy na základě stavu pointu se mění barva obrázku je nutné postupovat tak, že dva stavy obrázku (*true/false*) nahrajeme do platformy HAWK. Jestliže je odkázáno na soubory na PC kde je nainstalované prostředí, výsledný obrázek se nebude zobrazovat. Je nutnost nalinkovat tak soubor do adresáře HAWKu.

Pro přenos souborů do platformy slouží v navigačním okně platformy funkce *File Transform*. Zobrazí se okno, kde na levé straně je zobrazen lokální disk (je možné zvolit např. externí HDD apod.) a na pravé straně strom souboru v HAWKu. Možnost složky uložení je na volbě uživatele, v případě této práce byla zvolena složka v knihovně *Px/Klima,* jelikož se zde nacházejí další obrázkové nástroje a knihovny pro vizualizaci. Pomocí směrových šipek ve středu obrazovky jsou přenášeny soubory mezi okny.

Ve výsledné vytvořené vizualizaci je kompletní vizualizace modelů, včetně tlačítek a identifikační LED vytvořena mimo prostředí v grafickém softwaru Adobe Photoshop CC 2015.

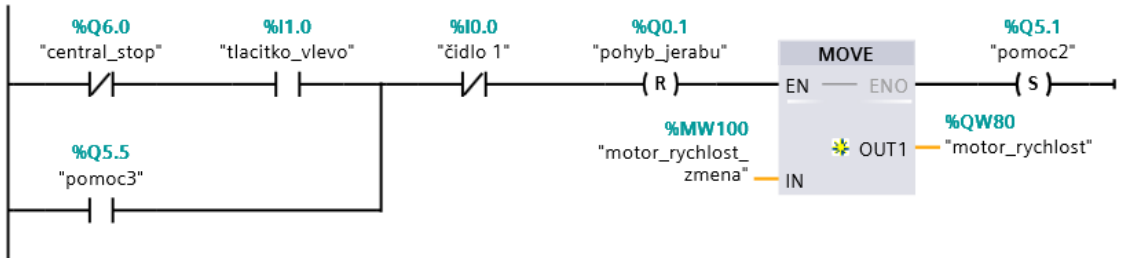
5.5 Logika spojená s ovládáním a vizualizací

Pro vytvoření vizualizace a částečného ovládání bylo nutné zasáhnout do zdrojového kódu automatu. Kód je naprogramován pro ovládání z panelu na pracovišti a jeho fyzickými tlačítky (vstupy I0.X). Pokud je chtíč ovládat model skrze prostředí, byla zvolena metoda, že ke každému fyzickému tlačítku je přidán

paralelně prvek pomocné proměnné. Stav proměnné bude měněn skrze prostředí a tím bude docházet k simulaci tlačítka.

Po určité době se tlačítko se sepnutého stavu rozepne, aby docházelo k simulaci reálného tlačítka a předcházelo se stále aktivnímu stavu.

Tímto způsobem se řeší veškeré prvky spojené s vizualizací procesu. Zobrazování polohy jeřábu je řešeno logikou podle aktivního stavu čidla a magnetu. Podmíněný proces aktivuje vždy jednu pomocnou proměnnou a podle ní se zobrazí obrázek jeřábu. Snímače polohy osy-x jsou boolean obrázek vázaný přímo na stav vstupu z automatu. Tímto způsobem jsou řešeny veškeré vstupy, tedy identifikační žárovky, bezpečnostní okruh a identifikace magnetu.



Obrázek 27 - Virtuální tlačítko (pomoc3) pro start hlavního motoru

Obrázek 27 popisuje možnost vidět realizaci virtuálního tlačítka *pomoc3*, tímto způsobem jsou realizovány prvky ovládacího panelu ve vizualizaci pro oba reálné modely.

Tlačítko je adresováno na výstup automatu Q5.5. Důvod je jednodušší a příjemnější přístup ke stavu proměnné a jednoduché adresování, než-li si předávat hodnotu pomocí holding registrů. Adresa je zvolena, aby nezasahovala do I/O modulu automatu, avšak pokud by byl připojen modul s dostatečným počtem výstupů, došlo by ke kolizi kódu a tím pádem jeho nefunkčnosti.

5.5.1 Standardizace rychlosti hlavního motoru

Podle manuálu k modelu jeřábu je nastavení rychlost hlavního (osa x) jeřábu realizováno vložením numerického čísla do proměnné QW80 a to v rozsahu 0-15000 [24]. Při uživatelské volbě je příjemnější volit rychlost v jednotkách uražené vzdálenosti za sekundu.

Postup byl takový, že byla 10-krát opakovaně měřena rychlost pohybu mezi dvěma senzory, aby byla minimalizována chyba měření. Jeřáb se u manuálního modelu zastavuje vždy na snímači ve stejné poloze. Měřený čas pro co nejpřesnější výsledek byl změřen pomocí časovače, který se aktivoval vždy když byla aktivní pomocná proměnná, která identifikuje pohyb jeřábu. Opakované měření bylo opět zprůměrováno a standardizováno pomocí přímky.

Vzdálenost mezi dvěma měřenými snímači byla 11,82 cm. Při vložení numerické hodnoty 10000 byl získán průměrný čas 3,58 s. Pro výpočet přímky je nutno získat čas pro jinou numerickou hodnotu. V tomto případě čas pro vstupní hodnotu 7000, která reprezentuje rychlost hlavního ramena motoru byla naměřena 7,35 s. Získaná výsledná rovnice přímky činí:

$$y = 568 * 10^{-6} * x - 2{,}366 \quad (1)$$

kde,
x - numerická hodnota prezentující rychlost (0-15000),
y - standardizovaný výstup v centimetrech za sekundu.

Následně byla pomocí matematických operací aplikována jednoduchá logika, kdy vstupním parametrem bylo numerické číslo prezentující rychlost motoru a na jeho výstupu sledovali rychlost motoru v cm za vteřinu. Vstupní číslo vetší jak 12000 spínalo bezpečnostní HW snímač, jelikož se nestihlo odvíjet lanko na navijáku.

5.5.2 Přístup k webovému prostředí

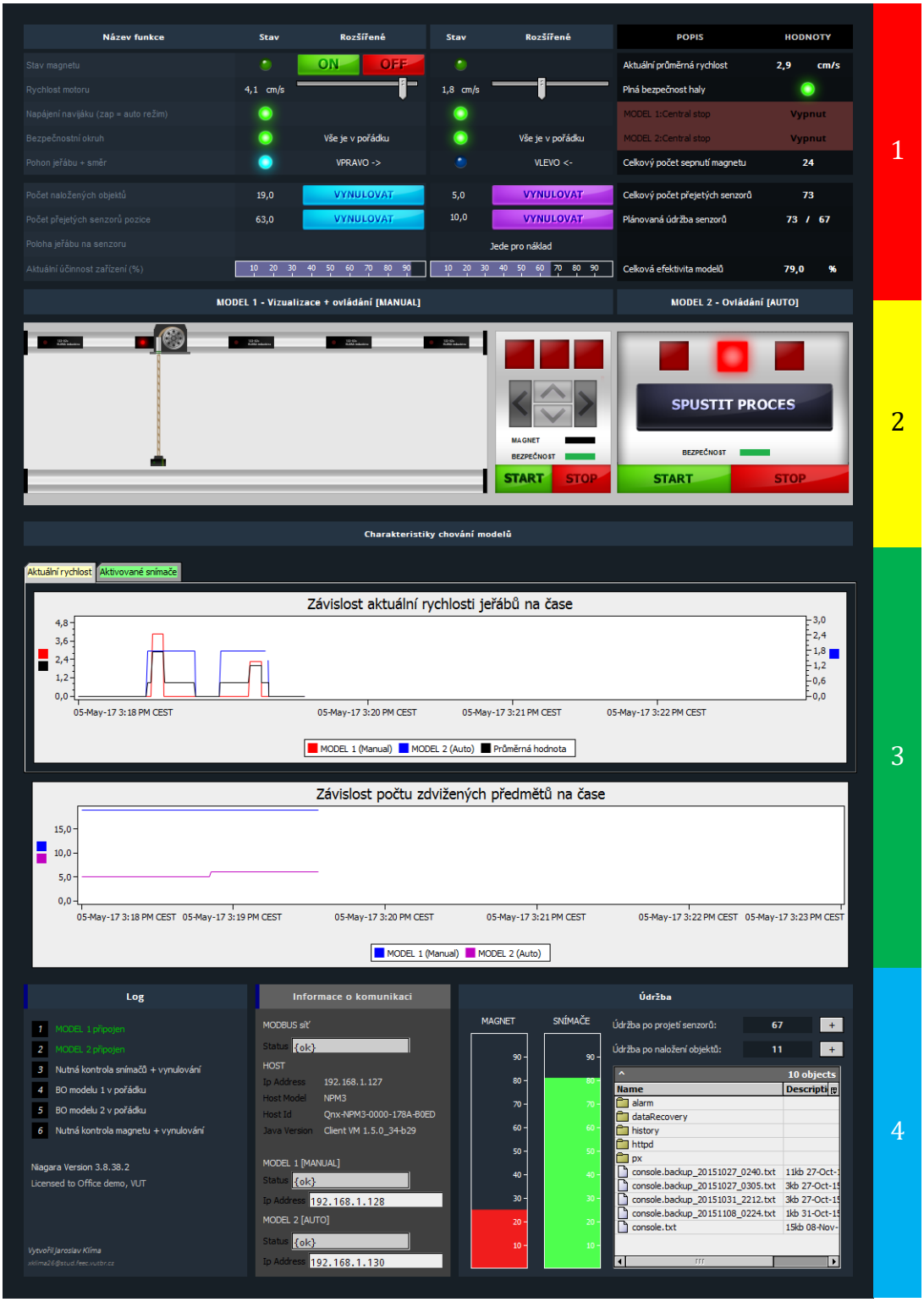
K vizualizované aplikaci je možnost se dostat ze zařízení disponující internetovým prohlížečem, nainstalovaným softwarem JAVA a připojením do stejné podsítě jako HAWK. Do kolonky internetové adresy je nutné zadat adresu regulátoru, příklad pro praktickou úlohu činní **http://192.168.1.127**. Ve stromu platformy je možnost nakonfigurování web serveru, avšak jeho výchozí nastavení je plně dostačující, takže bude ponecháno beze změny. Po načtení adresy se zobrazí přihlašovací okno do aktivní stanice v HAWKu, která může být vždy pouze jedna[2]. Přihlašovací údaje po připojení na regulátor jsou identické jako pro aktivní stanici v HAWKu:

- Přihlašovací jméno: *admin*
- Heslo: *Honeywell1*

Po úspěšném přihlášení se zobrazí identické prostředí jako při vytváření aplikace v prostředí COACH^{AX}. Zvolením složky *App* ve stromu stanice je otevřena požadovaná vizualizace.

5.6 Vizualizace pomocí webového prostředí

Na následujícím obrázku je ukázka prostředí (vizualizace) zobrazené ve webovém prohlížeči Mozilla Firefox v51.0 na operačním systému Windows 7 licence Home Premium 64bit.



Obrázek 28 - Kompletní vytvořená vizualizace skrze webové rozhraní

5.7 Popis vytvořené aplikace

Tato kapitola se bude věnovat podrobnému popisu možností a algoritmů aplikace na obrázku 28 vytvořené v inženýrském nástroji COACH^{AX} a zobrazení pomocí webového prohlížeče z PC.

5.7.1 Blok 1 - Okamžité hodnoty a stavy proměnných

Obsahuje informace o sepnutých snímačích a numerické hodnoty jako je statistika přenesených objektů a přejetých senzorů.

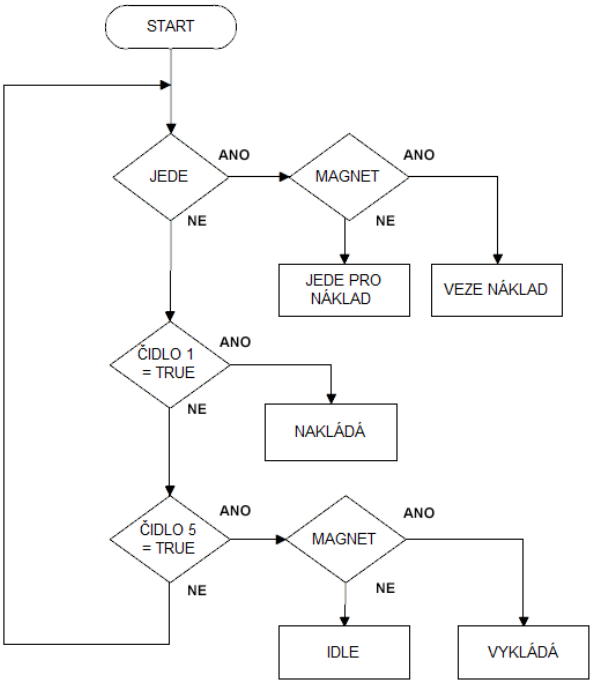
Obrázek 29 - Blok 1) Náhled na část vizualizace obsahující aktuální stavy proměnných

V prvním sloupci zleva jsou informace týkající se manuálního modelu automatu. Simulace diod (svítí, pokud je snímač sepnutý) je vázána obrázky přímo na stav proměnné, jedná se o tzv. *BooleanImage*, kde v nastavení parametrů je zvolen obrázek z adresáře regulátoru pro *TRUE* a *FALSE* stav.

Rychlost motoru lineárním posuvníkem je řešena tak, že je volena numerická hodnota na proměnnou hlavního motoru (kvůli bezpečnosti volen rozsah 6000-12000). Tato hodnota je přivedena na vstup provázané logiky podle vztahu (1). Zobrazovaná hodnota (standardizovaná pomocí přímky) je zobrazována v jednotkách cm/s. Přepočet je řešen pomocí logiky v příloze *B – 6*. Pohon motoru je rozlišen podle aktivního pointu (proměnné). Pokud je aktivní, hlavní motor jede vpravo z pohledu obsluhy. V opačném případě je identifikační LED neaktivní a pohon je nastaven na opačnou stranu. Ještě je důležité brát na zřetel, jestli vstupní numerická hodnota motoru není rovna nule. Pokud tomu tak je, pohon motoru není aktivní a přesto by v simulaci byla zobrazována rychlost. Tento problém je řešen logikou, pokud je vstupní parametr roven nule, rychlost motoru je bez přepočtu nastavena na nulu.

V manuálním režimu je nabízena možnost aktivování magnetu. Jde o základní prvek *button* z knihovny *kitControl* a v nastavení objektu je zvolena možnost *Actions -> Active*, tím je zapříčiněno, že po stisknutí je aktivován point, na který je navázán. Jako pozadí byl vybrán uživatelsky vytvořený obrázek.

Hodnoty stavu jeřábu a jeho status je řešen logikou, která pro lepší pochopení je prezentována pomocí vývojového diagramu (obrázek 30). Řešení v COACH^{AX} je v příloze B-5. *ČIDLO_1* prezentuje počáteční polohu hlavního ramene jeřábu. Jeho opakem je *ČIDLO_5*. Podmíněný krok *MAGNET* vyjadřuje stav magnetu na konci navijáku.



Obrázek 30 - Vývojový diagram pro identifikaci stavu modelu

Statistika ohledně počtu přejetých senzorů a přenesených předmětů je řešena pomocí virtuálního čítače v programu pro PLC a jeho stav je přenášen pomocí holding registru. Jeho reset vstup je boolean proměnná, která svojí aktivací restartuje čítač (proměnná obsažena v PLC kódu).

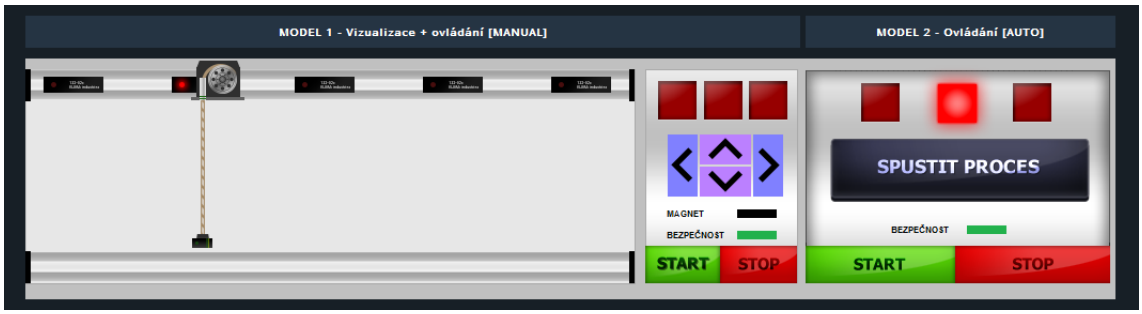
Central stop slouží k okamžitému zastavení modelu v jakékoliv poloze. Je to simulace kritické situace, kdy je nutnost všechny aktivní motory zastavit a přejít do krizové situace. Možnost central stopu je pouze z vizualizace, pokud dojde k aktivování central stopu, všechny motory modelu se zastaví a v logu i bloku 1 je možnost pozorovat upozornění na přerušení bezpečnostního okruhu a jeho identifikační LED změní barvu do červena. V tomto stavu je nemožné manipulovat s ovládáním motoru, až do situace, kdy obsluha vizualizace aktivuje tlačítko START, poté model pokračuje v činnosti, která předcházela aktivování central stopu.

Účinnost pracoviště je vyjádřena v procentech a ukazuje, na kolik procent je využívána rychlost hlavního motoru. Algoritmus výpočtu je takový, že aktuální

rychlost motoru podělíme jeho maximální rychlostí. Při maximální rychlosti je efektivita 100%. Celková efektivita je získána aritmetickým průměrem účinností obou modelů. Algoritmus v prostředí COACH^{AX} zobrazuje příloha B – 2.

5.7.2 Blok 2 - Vizualizace

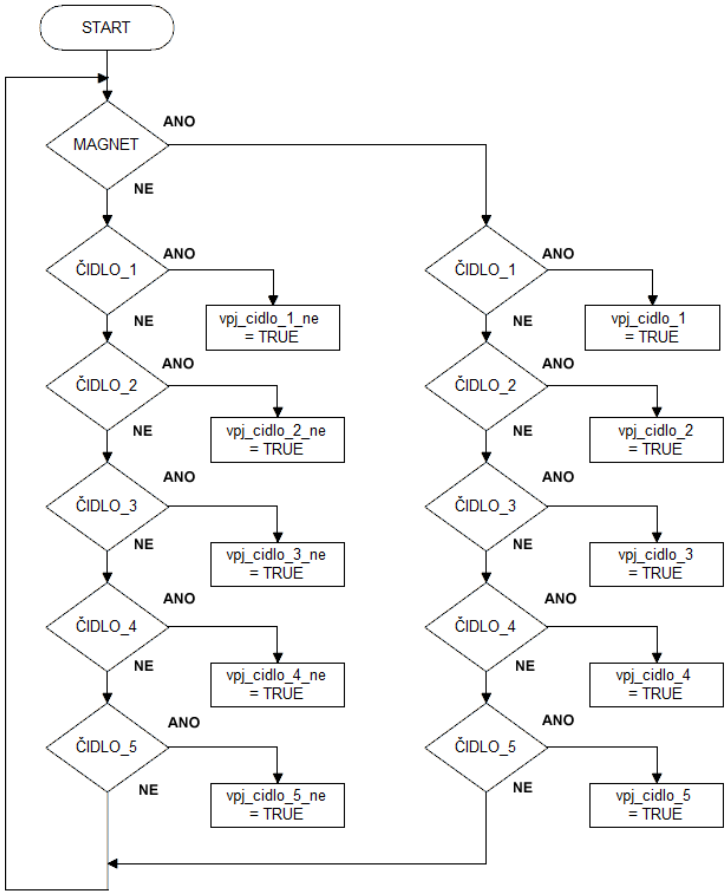
Tato část aplikace slouží pro vizualizaci reálného modelu. Na levé straně je model s manuálním ovládáním, který je kompletně vizualizován včetně ovládacího panelu.



Obrázek 31 – Blok 2) Vizualizovaný model jeřábu včetně ovládacích prvků

Vizualizovaný model odpovídá reálnému modelu, navíc s central stopem. Aktivování snímačů na hlavním rameni jeřábu, jak již bylo zmiňováno je pomocí *BooleanImage* s odkazem přímo na čidla, které jsou *Input* typem podle MODBUS tabulky (tabulka 1). Stejným způsobem byly řešeny identifikační žárovky. Obě krajní určují pohyb hlavního motoru jeřábu, prostřední světlo upozorňuje na dosažení konce ramena.

Poloha manipulátoru jeřábu, který se mění po dosažení každého z hlavních pěti čidel na hlavním rameni modelu je nastavována podle *TRUE* stavu operace *AND*. Vstupní hodnoty bloku jsou stav magnetu (bool) a stav čidla. Pro model vypnutého magnetu byly přidány možnosti s negovaným stavem magnetu. Pomocí aktivních čidel (vždy pouze jedno) a stavu magnetu byla možnost dosažení vždy pouze jednoho z deseti stavů. V závislosti na stavu *TRUE* jednoho z *AND* operandů byl zobrazen *BooleanImage*. Provázání algoritmu je v příloze *B – 8*, který je vyjádřen pomocí obrázku 32.



Obrázek 32 - Vývojový diagram pro selektování stavu modelu jeřábu při vizualizaci

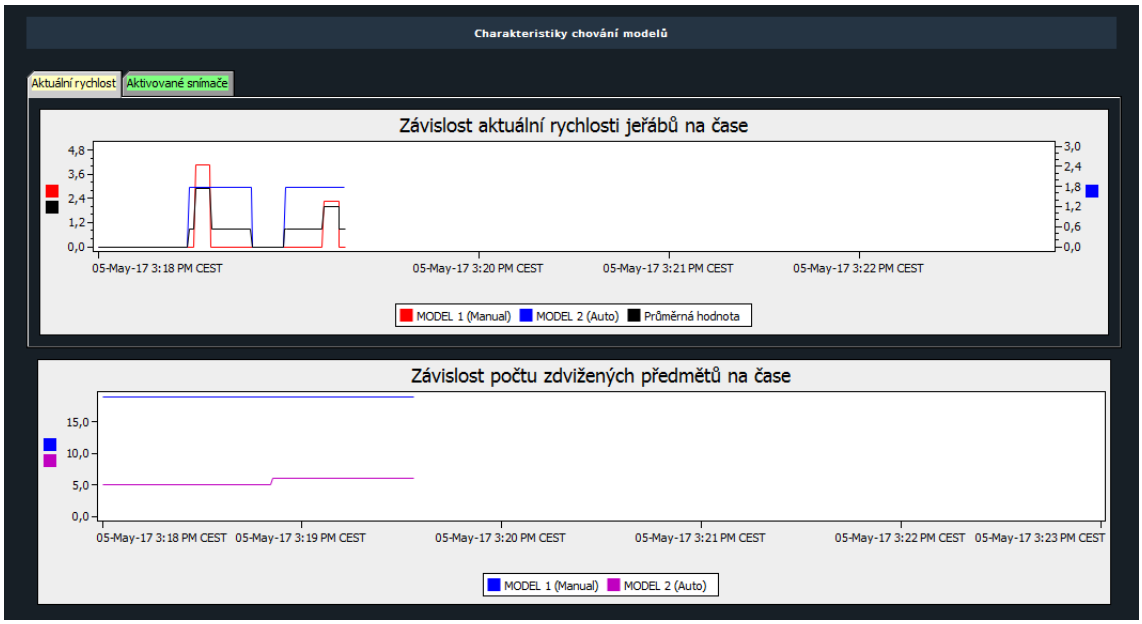
Oba modely mají implementované automatické navíjení navijáku, jelikož docházelo k potížím při dlouhodobém používání modelu. Lanko se zamotávalo do navíjecího bubenu nebo spínalo bezpečností HW čidlo umístěné na pohyblivém manipulátoru, který drží magnet. Při manuálním typu modelu je možné navijákem pohybovat po obou směrech osy y v každé situaci, tedy pokud není aktivovaný central stop.

Druhý ovládací panel označen MODEL 2 je automatický. Tlačítko SPUSTIT PROCES má stejný účinek jako jediné funkční fyzické tlačítko na reálném modelu umístěno uprostřed. Je umístěno paralelně k fyzickému tlačítku. Možnost virtuálního tlačítka je při inicializaci nastaveno na *Action -> Active*. V tomto případě je nutnost upravit možnost *activePermanent* na hodnotu 250ms, jelikož je nežádané proces opakovat v nekonečném cyklu. Proto se po čase 250ms bude tlačítko automaticky rozepnuto a tak dosaženo simulace reálného tlačítka. Krajní žárovky identifikují směr hlavního ramena a prostřední světlo identifikuje stav magnetu, při jeho aktivním stavu žárovka svítí.

Grafika byla kompletně vytvořena mimo prostředí COACH^{AX}, abychom dosáhli co nejvěrohodnější kopie reálného modelu, vytvořené obrázky pak bylo nutné importovat do adresáře regulátoru.

5.7.3 Blok 3 – Grafické vyjádření modelů

Prostředí podporuje ve výchozí konfiguraci časovou závislost pointu (proměnné). Graf byl vytvořen přesunutím pointu na prostředí v režimu grafické úpravy (*Edit Grahics).* Po přesunutí byla zobrazena možnost inicializované proměnné, možnost Time Plot. Velikost okna s grafem lze přizpůsobovat pomocí šipek na krajích okna. V první části bloku je k dispozici přepínatelná lišta dvou grafů. Pro její realizaci bylo nutné přidat do prostředí knihovnu *Chart*.



Obrázek 33 - Blok 3) Časové závislosti nejsledovanějších veličin z reálných modelů

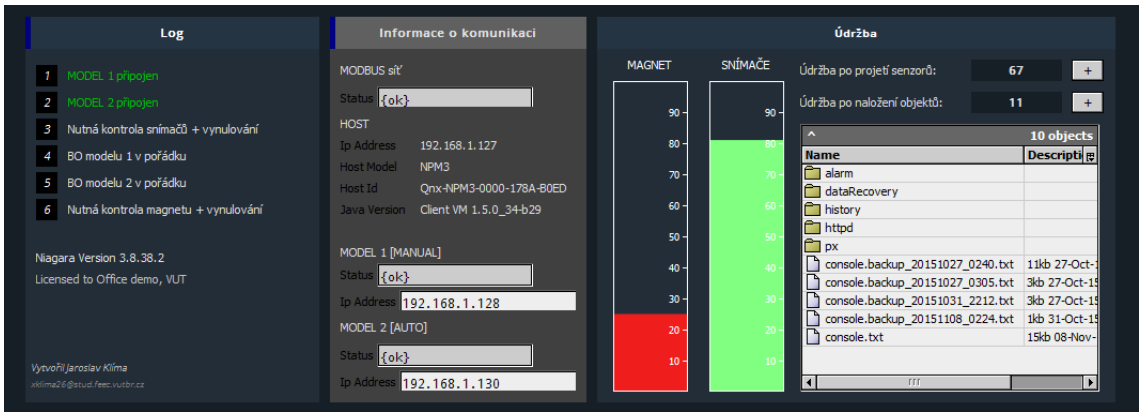
Knihovnou *control* inicializujeme přepínací lištu a pomocí knihovny *chart* byly přidány přesunutím do vrstvy (pravá strana grafického editoru) kartu. Ve stejné knihovně byla přidávána další proměnná do jednotlivých grafů (příklad dvou grafických závislostí v jednom grafu). V nastavení proměnné (dvojité kliknutí v možnostech vrstev) je možnost nastavení např. tloušťky čáry nebo meze hodnot.

Průměrné hodnoty jsou získány pomocí aritmetického průměru hodnot z obou modelů, realizované blokem *Average* ze základní knihovny *kitControl*. *Average* blok, který je po inicializaci zobrazen mezi pointy nebo ve složce tvořeného algoritmu se přesune do vrstvy grafu pro dosažení závislosti do jednoho grafu.

V používané verzi prostředí (3.8.38.2) je možnost přenést data z grafů do souborů typu *.csv, *.pdf a *.html a to kliknutím pravým tlačítkem myši na grafickou oblast a zvolení možnost *Export data*. Verze 4.0 a výš podporuje přímo funkci automatického ukládání a čtení dat do formátu *.csv [26].

5.7.4 Blok 4 – Doplňující informace

Na levé straně bloku se nachází log informací, co se v daném momentu děje s modely.



Obrázek 34 - Blok 4) Doplňující informace k vizualizaci, komunikaci s modely a regulátorem HAWK

Texty jsou zobrazovány na základě bool pointů vytvořené v prostředí COACH^{AX} a na jejich stavu následně zobrazené *Facets*. Pro příklad hláška *BO v pořádku* je výstup central stopu a pointu pro bezpečnostní okruh daného modelu. *Facets* jak již bylo zmiňováno v předchozí kapitole, jsou úpravy pro výrazy stavu bool proměnné. Ve výchozím nastavení jsou pro pravdivý stav *TRUE* a pro nepravdu *FALSE*. Řetězce tedy byly přizpůsobeny pro aktuální úlohu.

Informace o komunikaci byly získány přetažením celého modelu automatu v nastavení (*Config)* stanice na grafický pracovní režim. V nastavení proměnné volba možnosti *Properties*. Zde je získán přístup do nastavení celého modelu odkud je možnost brát informace, které volíme při inicializaci zařízení jako je IP adresa, status připojení nebo stav alarmů. Informace ohledně verze nainstalované verze JAVA, verze prostředí COACH^{AX} nebo hostovi (HAWK) zobrazíme v nastavení stanice (dvojité kliknutí na aktivní stanici v platformě) v sekci *Summary Properties*.

Pravá část bloku se věnuje údržbě. Na základě počtu sebraných informací z modelů se po určitém počtu sbírání dat, který je možný nastavovat ve spodní části *tlačítkem plus,* aktivuje stav, který vyžaduje nulování snímačů a je prezentován příkazem obsluhy v logu k demonstračnímu zkontrolování snímačů. Nastavení meze je editovatelný numerický point, který je porovnáván se součtem

počtu použití magnetu nebo přejetých senzorů. Při překročení nastaveného limitu přestane grafická závislost (sloupcové grafy) zobrazovat aktuální údaj a log vyžádá kontrolu s podrobnějšími informacemi. Logika spočívá v porovnávání součtu hodnot z modelů ke zvolenému pointu. Zobrazení v části logu bylo řešeno pomocí vytvořené bool proměnné s upravenými *Facets* (viz přechozí kapitola), v tomto případě se aktivuje bool proměnná při dosažení zadaných limit. Řešení je zobrazeno v příloze B-3 a B-4.

Pro okno s adresářem byl použit souborový prohlížeč důvod přiložení je z informativního prohledávání uživatelsky vytvořených souborů a celkového adresáře stanice (včetně veškerých doinstalovaných a importovaných knihoven z platformy), tzn. pro ukázku uložených uživatelských souborů.

6 ZÁVĚR

Výsledkem této práce je nadřazený systém oproti vrstvě programovatelných automatů, neboli aplikace pro sbírání údajů, statistiku a vizualizaci dat z reálných modelů portálového jeřábu, řízených automaty S7-1200 od firmy SIEMENS.

V první části práce je věnována seznámení s vývojovým prostředím od firmy Centraline/Honeywell jménem COACH^{AX} a HAWK regulátorem dostupným v laboratoři moderních metod v automatizaci. Práce je literární rešerší pro komunikační možnosti regulátoru v základní konfiguraci a jeho možných modifikací. Druhá kapitola přibližuje nejčastěji používané aplikace, knihovny, základní funkce a nástroje vedoucí k vytvoření aplikace.

Zadání požaduje demonstraci komunikaci po zvoleném komunikačním protokolu MODBUS TCP/IP, který byl prezentován pomocí jednoduché úlohy, kterou byla změna výstupu automatu a přenesení pomocné proměnné vyžadující práci s holding registry. Funkčnost byla demonstrována pomocí výstřižku obrazovky před a po změně parametrů skrze prostředí COACH^{AX}. Reálné modely jsou v postavení serverů a odesílají data ke klientovi, kterým je HAWK. Tento síťový model byl zvolen z důvodů jednoduššího konceptu sítě, kdy se klient dotazuje serverů na stavy nebo hodnoty proměnných.

V praktické části byly vytvořeny dva rozdílné zdrojové kódy pro identické modely jeřábů řízeny pomocí PLC S7-1200 v prostředí TIA PORTAL verze 13. V prostředí COACH^{AX} byla vytvořena ze sbíraných dat vizualizace modelu, statistiky jako počet přejetých senzorů a celkové efektivity modelů. Při vytváření se vyskytly problémy ohledně rozšíření JAVA. Problém řešila aktualizace. Dalším překázkou byly výchozí knihovny v prostředí. K plné realizaci vizualizace byly zapotřebí funkce, které dostupné knihovny neobsahovaly. Skrze nástroj v platformě s názvem *Software manager* bylo nutné doinstalovat potřebné rozšíření. Pomůckou při řešení problémů bylo oficiální fórum nástroje COACH^{AX} dostupné v anglickém jazyce.

Práce může sloužit jako výukový materiál pro propojení periférií přes komunikační protokol MODBUS na TCP/IP nebo popis vytvoření aplikace v prostředí COACH^{AX}. S drobnými modifikacemi je možno model nakonfigurovat pro sběr dat z různých průmyslových reálných jeřábů, dopravníků a teplotních systémů. Modifikací se rozumí bezpečnostní prvky a požadavky zadavatele.

Výsledná vizualizace byla realizována pomocí webového prostředí. Jenž je přístupné ze zařízení, které je vybaveno webovým prohlížečem a je ve stejné podsíti jako regulátor. Vytvořená aplikace je plně funkční a opakovaně testována pomocí ovládacích prvků jak na reálném modelu z pohledu operátora, tak aplikaci ve webovém rozhraní. Aplikace je testována na různé kombinace stavů, které

mohou nastat, a tak je ošetřena proti nežádoucím stavům, jako je zastavení posuvné části jeřábu mezi dvěma snímači centrálním stopem, a poté následný dojezd na pozici před zastavením.

LITERATURA

[1] *CentraLine* [online]. Honeywell Inc., 2015 [cit. 2017-02-25]. Dostupné z WWW: http://products.centraline.com/cz/index.html

[2] HAWK: *regulátor a integrátor pro aplikace automatizace budov*. *products.centraline.com* [online]. Honeywell Inc., 2015 [cit. 2017-01-30]. Dostupné z WWW: http://products.centraline.com/cz/ecatdata/pg_clhawk.html

[3] MATZ, Václav. *Typy sběrnic a protokolů používaných pro komunikaci systémů* [online]. s 26 [cit. 2017-03-01]. Dostupné z WWW: http://www.ib.cvut.cz/sites/default/files/Honeywell_prednasky/Protokoly_1.pdf

[4] CentraLine. HAWK DATA SHEET [online]. [cit. 2017-03-01]. Dostupné z WWW: http://www.stavebnivyrobekroku.cz/db_binary_file/other/803

[5] VOJÁČEK, Antonín. *Analogový vs. digitální přenos hodnot. Kdy ještě volit analogový výstup?* [online]. 2015 [cit. 2017-03-01]. Dostupné z WWW: http://automatizace.hw.cz/mereni-a-regulace-prumyslove-sbernice-a-komunikace/analogovy-vs-digitalni-prenos-hodnot-kdy-jeste-volit-analogovy-vystup

[6] HONEYWELL. *Coach AX (Level 1) Training: Creating a LON – BacNet Integration*. Tréninkový materiál.

[7] HONEYWELL. *CLAXHAWKIO34, IO-34 module for HAWK* [online]. Germany: Honeywell, 2008 [cit. 2017-03-10]. Dostupné z WWW: http://products.centraline.com/ru/pdf/en1z0947- ge51r0608.pdf

[8] M-Bus [online]. České vysoké učení technické v Praze , Fakulta elektrotechnická, 2008 [cit. 2017-03-10]. Dostupné z WWW: http://measure.feld.cvut.cz/node/788

[9] JARKA, Jaroslav. OPC komunikace a Wonderware software [online]. Pantek, 2012 [cit.2017-03-10]. Dostupné z WWW http://www.pantek.cz/pdf/ostatni/120919_OPC_komunikace.pdf

[10] KUNC, Josef. *ABB: KNX/EIB Příklady sběrnicovách systémů* [online]. 2008 [cit. 2017-03-10]. Dostupné z WWW: http://elektrika.cz/data/clanky/abb-

systemove-elektricke-instalace-knx-eib-2013-3-cast/view?searchterm=KNX/EIB

[11] VOJÁČEK, Antonín. *Úvod do BACnetu: Building Automation and Controls Network* [online]. 2012 [cit. 2017-03-10]. Dostupné z WWW: http://automatizace.hw.cz/uvod-do-bacnetu-building-automation-and-controls-network

[12] RONEŠOVÁ, Andrea. *Přehled protokolu MODBUS* [online]. 2005 [cit. 2017-03-10]. Dostupné z WWW: http://home.zcu.cz/~ronesova/bastl/files/modbus.pdf

[13] HONEYWELL. *CentraLineAX*. Tréninkový materiál.

[14] AUTOMA. *Novinky ve vývojovém prostředí TIA PORTAL* [online]. 2015 [cit.2017-03-14]. Dostupné z WWW: http://automa.cz/cz/casopis-clanky/novinky-ve-vyvojovem-prostredi-tia-portal-2015_03_53519_8016/

[15] MATZ, Václav. *Nová generace řídicích systémů Honeywell CentraLineAX*. [cit. 2017-03-14]. Dostupné z WWW: https://ib.cvut.cz/sites/default/files/Honeywell_prednasky/3a_prezentace.pdf

[16] HONEYWELL. *RS-232 option card for Hawk*. [cit.2017-03-18]. Dostupné z WWW: http://products.centraline.com/nl/pdf/en1z0949-ge51r0608.pdf

[17] HONEYWELL. *RS-485 option card for Hawk*. [cit.2017-03-18]. Dostupné z WWW: http://products.centraline.com/cz/pdf/en1z0950-ge51r0615.pdf

[18] REAL CONTROL SOLUTIONS. *CoachAX and Graphics* [online]. 2012 [cit. 2017-03-18]. Dostupné z WWW: http://realcontrolsolutions.co.uk/learn/honeywell/coach-ax-and-graphics/

[19] REAL CONTROL SOLUTIONS. *CoachAX and Graphics Part II* [online]. 2012 [cit. 2017-03-18]. Dostupné z WWW: http://realcontrolsolutions.co.uk/learn/honeywell/coach-ax-and-graphics-part-ii/

[20] REAL CONTROL SOLUTIONS. *CoachAX and Alarms* [online]. 2012 [cit. 2017-03-18]. Dostupné z WWW: http://realcontrolsolutions.co.uk/learn/tag/coachax/page/3/

[21] FOXCON. *Co je OPC? OPC server? OPC klient?* [online]. [cit. 2017-03-20]. Dostupné z WWW: https://www.foxon.cz/cs/blogs/80-co- je-opc-opc-server-opc-klient-.html

[22] [cit. 2017-03-10]. Dostupné z WWW: http://www.distech-controls.com/en/ca/products/field-devices/nc-network-connectivity/related-products/lon-pci-cards/

[23] [cit. 2017-03-10]. Dostupné z WWW: http://www.distech-controls.com/en/as/products/field-devices/nc-network-connectivity/related-products/usb-network-interface/

[24] ŠTOHL, JIRGL, ARM, MIŠÍK. *Manuál k předmětu BPPA*. Brno: FEKT BUT, 2017 [cit.2017-03-10].

[25] PLC AUTOMATIZACE. *Typy dat* [online]. [cit.2017-03-10]. Dostupné z WWW: http://plc-automatizace.cz/knihovna/data/data-typy-dat.htm

[26] NIAGARA COMMUNITY. *Forum: Scheduling Exports of History CSV* [online]. 2009 [cit.2017-03-20]. Dostupné z WWW: https://www.niagara-community.com/Comm_ChatterAnswers?id=906D0000000TYxtIAG

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

TCP/IP	-	Transmission Control Protocol/Internet Protocol
HVAC	-	Heating, ventilation and air conditioning
VDC	-	Voltage Direct Current
LON	-	Local Operating Network
OPC	-	OLE for Process Control
M-BUS	-	Meter-Bus
EIB	-	European Installation Bus
PDU	-	Protocol Data Unit
ADU	-	Application Data Unit
UTP	-	Unshielded Twisted Pair
HR	-	Holding register
RW	-	Read-Write
HW	-	Hardware
SW	-	Software
PLC	-	Programmable Logic Controller
CRC	-	Cyklický redundantní součet
LRC	-	LyRiCs
CR	-	Carriage return
LF	-	Line feed
OS	-	Operační systém
I/O	-	Input/Output (vstup/výstup)
JVM	-	Java Virtual Machine
HDD	-	Hard disk drive (pevný disk)
TIA	-	Totally Integrated Automation
PC	-	Personal Computer
RAM	-	Random Access Memory
NAPŘ	-	Například
TZN	-	To znamená
TZV	-	Tak zvaný
BPPA	-	Prostředky průmyslové automatizace

SEZNAM PŘÍLOH

Příloha A. Obsah přiloženého CD
Příloha B. Zdrojové kódy z prostředí COACH^{AX}
Příloha C. Fotodokumentace

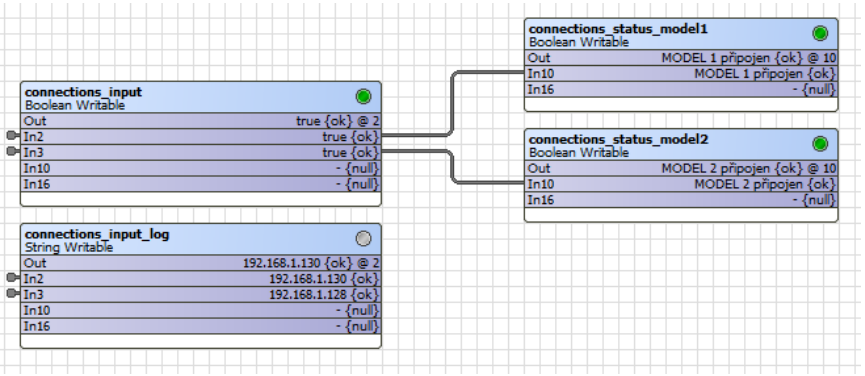
PŘÍLOHA A – Obsah přiloženého CD

- BP_APLIKACE_S_HAWK_REGULATOREM_KLIMA_2017.pdf
- BP_PRILOHA_PLC_A_01.pdf – Model 1 PLC, MODBUS inicializace
- BP_PRILOHA_PLC_A_02.pdf – Model 1 PLC, zdrojový kód
- BP_PRILOHA_PLC_B_01.pdf – Model 2 PLC, MODBUS inicializace
- BP_PRILOHA_PLC_B_02.pdf – Model 2 PLC, zdrojový kód
- BP_PRILOHA_PLC_B_03.pdf – Model 2 PLC, hlavní strom zdrojového kódu

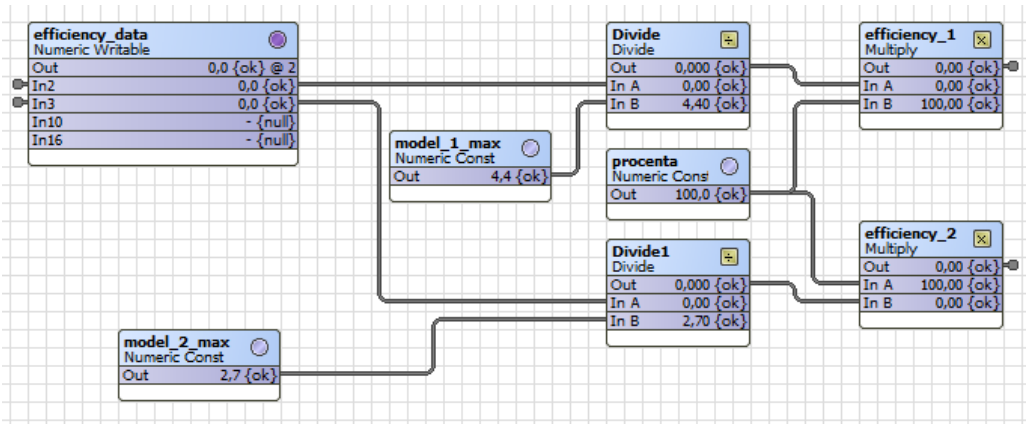
- BP_PRILOHA_PLC.zip – Soubory projektů z prostředí TIA PORTAL V13
- BP_PRILOHA_HAWK_STANICE.zip – Souborový strom projektu z COACH^{AX}
- BP_FOTOGRAFIE.zip – Nafocené fotografie pracoviště

PŘÍLOHA B – Zdrojové kódy

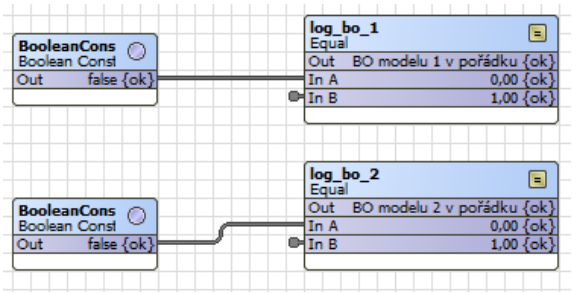
B – 1 Blok Connections



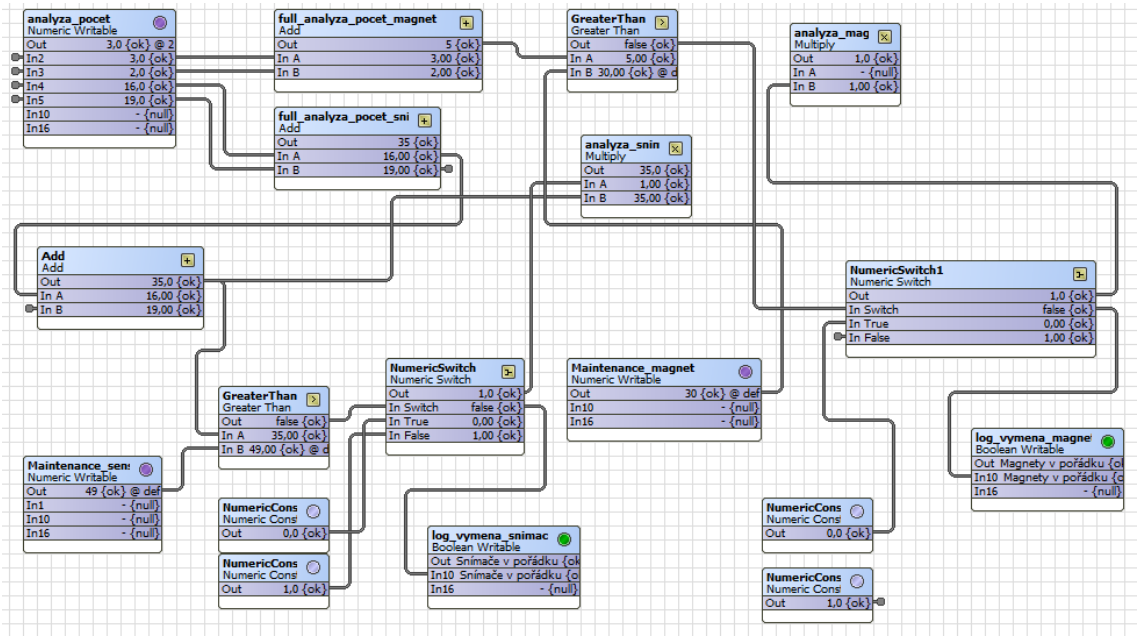
B – 2 Blok Efficiency



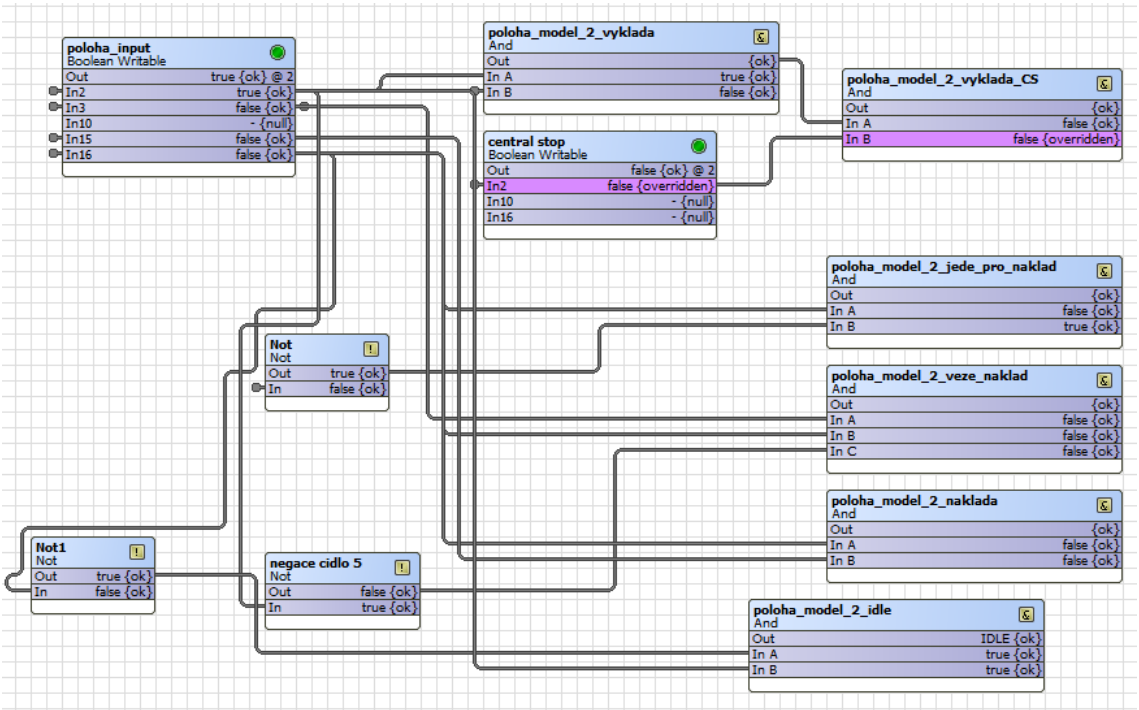
B – 3 Blok Maintenance (část 2 – bezpečnostní okruhy)



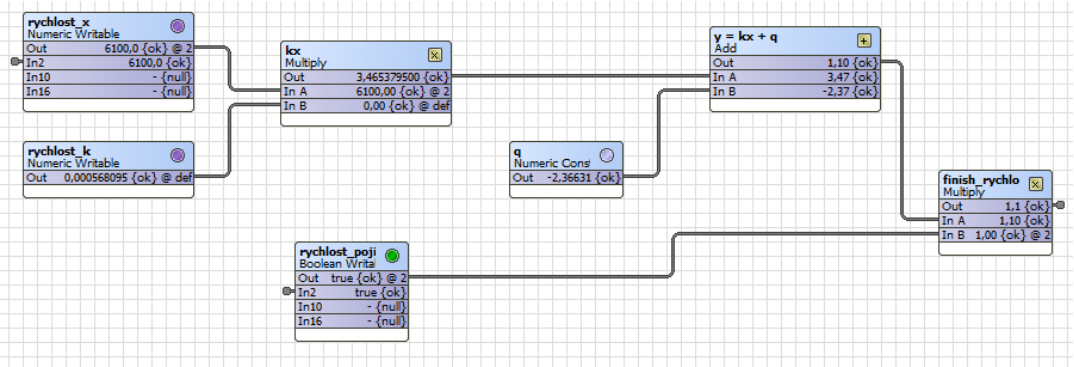
B – 4 Blok Maintenance (část 1)



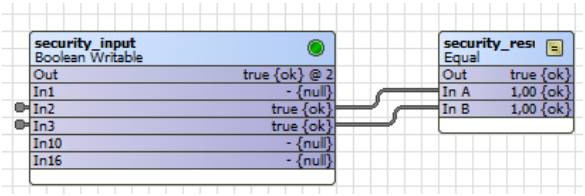
B – 5 Blok poloha_model_2



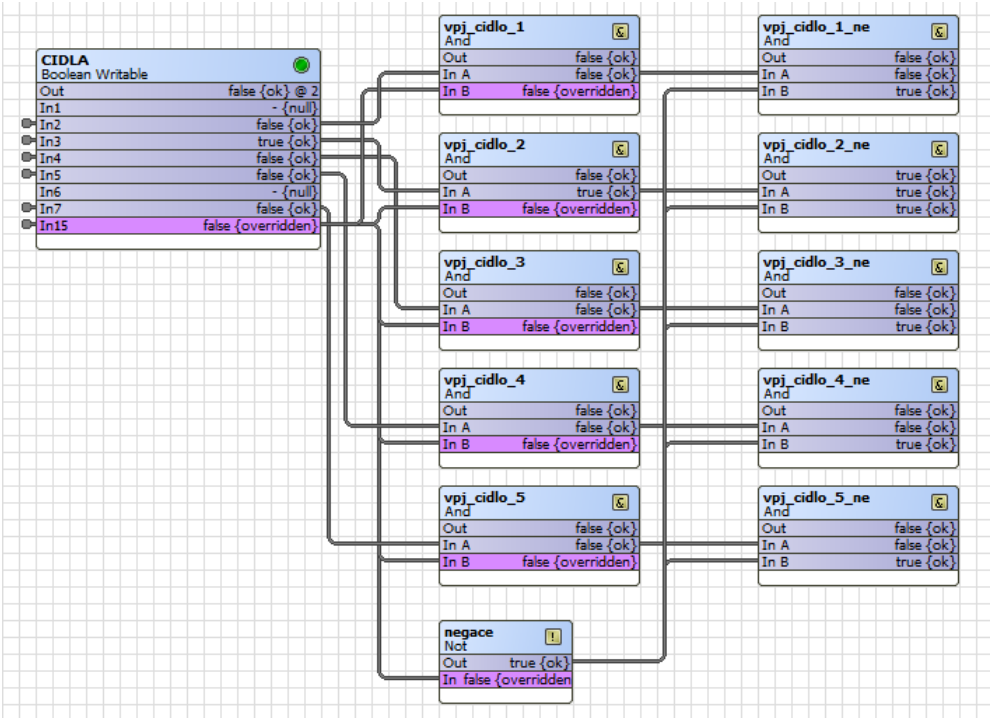
B – 6 Blok rychlost_motoru_standardizovane



B – 7 Blok security

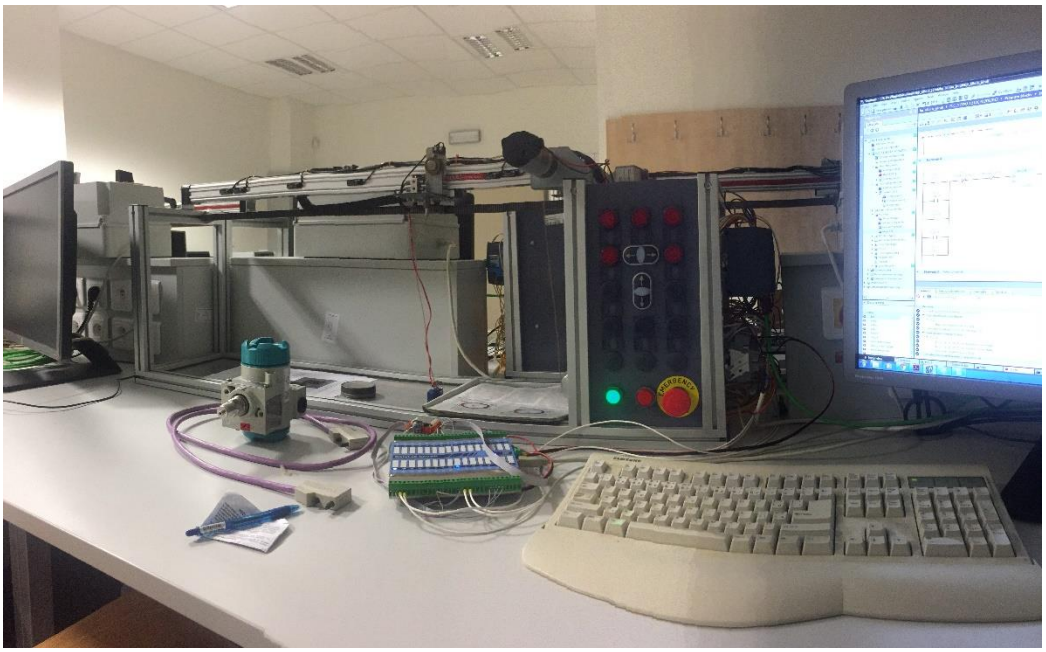


B – 8 Blok vizualizace_poloha_jerabu



PŘÍLOHA C – Fotodokumentace

C – 1 Model PLC (2x)



C – 2 Pracoviště s HAWK regulátorem a modulem IO34

C – 3 Celkový vzhled pracoviště